Stochastic Inventory Routing with Time-based Shipment Consolidation

Danja R. Sonntag  
Business School, University of Mannheim, Germany  
dsonntag@mail.uni-mannheim.de

Albert H. Schrotenboer  
Operations, Planning, Accounting, and Control Group, School of Engineering, Eindhoven University of Technology, Netherlands  
a.h.schrotenboer@tue.nl

Gudrun P. Kiesmüller  
TUM School of Management, TUM Campus Heilbronn, Technical University of Munich, Germany  
gudrun.kiesmueller@tum.de

Inspired by the retail industry, we introduce a fundamentally new approach towards stochastic inventory routing by replenishing retailers from a central warehouse using a time-based shipment consolidation policy. Such a time-based dispatching policy, where retailers facing stochastic demand are repetitively replenished at fixed times, is essential in practice. It allows for easy incorporation with dependent up- and downstream planning problems such as personal staffing and warehouse operations, and has become a standard part of transportation contracts. We provide a new chance-constrained model that determines an optimal clustering of retailers in groups, their associated routing and shipment interval, and each retailers’ optimal inventory level. A newly developed branch-and-price-and-cut algorithm solves our model to optimality. Its efficiency comes from a tailored labeling algorithm for solving the pricing problem that relies, among others, on an optimality pruning criterion based on the approximate solution of a 0,1-knapsack problem. Computational experiments show that our exact method can solve instances of up to 60 retailers to optimality. Besides, we accommodate practitioners by providing fast heuristics and a hybrid heuristic that provides excellent solutions with an optimality gap of less than 1%. Finally, we show that incorporating uncertainty already in the planning process is essential for stochastic inventory routing with time-based shipment consolidation, as it results in overall cost-savings of 7.7% compared to the current state-of-the-art.

Key words: stochastic inventory routing; transportation; inventory control; time-based shipment consolidation; retail; branch-price-and-cut; chance constraints

1. Introduction

How to optimally replenish inventory at geographically dispersed retailers from a central warehouse is a fundamental question at the interface of inventory management and transportation. At this interface, operations managers need decision support to coordinate when shipments are sent to retailers, how much will be sent to each retailer, and how these shipments are consolidated in multiple vehicle routes to minimize inventory holding and transportation costs. Practical applications by Duffy (2004), Gaur and Fisher (2004), Coelho and Laporte (2014), and Van Anholt et al. (2016) have shown that an integrated view on inventory management and transportation leads to significant cost savings in practice. In particular, Van Anholt et al. (2016) estimated the expected business cost savings at about €10.1 million per year for a Dutch ATM
operator, and Gaur and Fisher (2004) reported cost-savings of 4% during the first year in a study with Albert Heijn, a leading supermarket chain in the Netherlands.

For planning the retailer’s inventory replenishments, it is necessary that shipments leave the warehouse and arrive at the retailers at fixed and predetermined points in time to allow for coordination with dependent up- and downstream planning processes such as material handling and staffing. The necessity for such time-based dispatching policies has been exemplified by case studies from Holzapfel et al. (2016) at a major European grocery retailer, Stenius et al. (2016) at a European metal sheet production company, and Campelo et al. (2019) for medicine distribution in the pharmaceutical sector. Time-based dispatching policies are also beneficial for logistics service providers. Namely, drivers can be utilized more efficiently because they become familiar with their routes, and, on a higher level, logistics providers can better plan the vehicles and personnel required. Consequently, companies can enter cheaper long-term contracts with logistics service providers (Gaur and Fisher 2004).

The resulting inventory replenishment strategy, where shipments are dispatched from the warehouse at fixed points in time to replenish inventory at predetermined groups of retailers, is known in the literature as a time-based shipment consolidation policy (see, e.g., Higginson and Bookbinder 1995, Marklund 2011, Ülkü and Bookbinder 2012, Stenius et al. 2016, Johansson et al. 2020). Optimizing inventory replenishments under time-based shipment consolidation is challenging because retailers face stochastic demand. In such a case, each replenishment takes place at fixed times while the amount to be shipped is stochastic. In the literature, the joint optimization of inventory replenishments and transportation is known as the inventory routing problem (Coelho et al. 2014b). However, researchers have been studying time-based shipment consolidation policies only when retailer demand is deterministic. Alternatively, researchers studying inventory routing problems under stochastic retailer demand focus on completely dynamic replenishments, meaning that despite replenishment quantities being stochastic, retailers are not visited at fixed points in time. There is no study integrating time-based shipment consolidation policies in stochastic inventory routing.

In this paper, we incorporate time-based shipment consolidation into stochastic inventory routing by proposing a new, generic chance-constrained model that consists of the following key elements. First, we assign retailers to fixed groups with fixed shipment intervals, such that all retailers of a single group are replenished jointly at fixed points in time. Second, we include a service level constraint at each retailer and, thus, optimize both cycle stock and safety stock. Third, we include emergency shipments in case truck capacity is insufficient. Our model minimizes the expected per-period transportation, emergency, and holding costs by jointly optimizing the retailer groups, the associated shipment intervals and vehicle routes, and each retailer’s cycle and safety stock. The models’ complexity regarding the joint optimization of all decision variables stems from the strong connection between all decisions. Namely, the quantity of cycle and safety stock depends on the length of the shipment interval, which depends on the composition of the retailer groups.
To solve our chance-constrained optimization model, we transform it into an integer program for which we present a tailored branch-price-and-cut algorithm. To solve the associated pricing problem, we introduce a new pruning criterion based on the approximate solution of a 0,1-knapsack problem. The approach appears to be efficient because it solves instances of up to 60 retailers to optimality. Notably, this is the first exact solution approach to the stochastic inventory routing problem that considers the joint optimization of retailer clusters, their corresponding shipment intervals, the routing within each cluster, and the base stock levels at all retailers. Moreover, we provide tailored constructive heuristics that also serve as input to the branch-price-and-cut algorithm and a hybrid heuristic that solves all considered problem instances with an optimality gap of typically less than 1% in negligible time. We show that including the uncertainty of retailer demand in the planning process is crucial for controlling costs and capacity utilization. Relying on expected demand and buffer space in trucks yields an average cost increase of 7.7% and insufficient truck capacity in about 95% of the considered instances, despite that we re-optimize shipment intervals to prevent this to the best extent possible. This shows that our new approach, not suffering from this drawback, determines structurally different replenishment schemes that can better mitigate retailer demand uncertainty by making more efficient use of the available truck capacity.

In summary, the contributions of this paper are as follows: (1) We present a novel approach and an associated new, generic model to decision making within the field of stochastic inventory routing problems by imposing time-based shipment consolidation policies that ensure replenishments at fixed times as requested by many companies for the ease of coordination. Therefore, we optimize not only transportation cost and cycle stock but also safety stock and emergency shipment costs. (2) We develop a tailored branch-price-and-cut algorithm to provide optimal solutions. (3) We provide tailored constructive heuristics and a hybrid heuristic that exhibit excellent performance in negligible time. (4) We show the importance of including uncertainty in jointly optimizing inventory replenishment and transportation decisions under time-based shipment consolidation policies.

The remainder of the paper is organized as follows. In Section 2, we describe the problem in detail. In Section 3, we review the relevant literature from the fields of inventory management, capacitated clustering, and inventory routing, and emphasize the research gap that this paper is closing. Based on the problem statement in Section 2, we formulate a new, integer chance-constrained optimization model in Section 4. In Section 5, we present an exact solution approach and in Section 6 several heuristics. In Section 7, we provide numerical results showing the performance of all solution approaches, the importance of considering uncertainty, and insights into the structure of the optimal solution. We conclude the paper and provide an outlook on future research opportunities in Section 8.

2. Problem Statement

We consider a periodic, infinite-horizon, single-warehouse multiple-retailer distribution inventory system as depicted in Figure 1. In the following, we provide a high-level description of our system and postpone the
formulation of our optimization problem to Section 4. For an overview of the main notation used, we refer to Appendix A.

Figure 1  Single-Warehouse Multiple-Retailer Distribution Inventory System with Shipment Consolidation

The system consists of a set of retailers $\mathcal{N} = \{1, \ldots, N\}$, where each retailer $i \in \mathcal{N}$ faces stationary but stochastic period demand $D_i$ for a single item with known distribution and known expectation $\mathbb{E}[D_i]$ and variance $\text{VAR}[D_i]$. Note that a single item model is suitable even in a retail setting with thousands of products, if the demand is aggregated to, e.g., units of volume as it is done by Albert Heijn (Gaur and Fisher 2004). Demand occurring at each retailer is either satisfied from stock on hand immediately or backlogged in case of stock-outs. Backorders are satisfied as soon as possible on a first-come, first-served basis once a new delivery arrives at the retailer. Each retailer $i$ imposes a target service level $\alpha_i^*$ that refers to the non-stock-out probability at retailer $i$, which ensures a high level of customer satisfaction. Notice that under high service requirements at the retailers, adopting lost sales instead of backorders does not provide further insights on how time-based shipment consolidation policies can be incorporated in stochastic inventory routing but only further complicates the model. We leave the consideration of lost sales for future research.

To replenish stock from the central warehouse, each retailer uses a periodic replenishment policy. More precisely, each retailer $i$ places orders according to an $(R_i, S_i)$-policy comprising a review period $R_i$ and a base-stock level $S_i$. Under an $(R_i, S_i)$-policy, retailer $i$ places an order every $R_i$ periods such that the inventory position after ordering is equal to the base-stock level $S_i$. We denote the vector of base-stock levels for all retailers $i \in \mathcal{N}$ by $S = (S_1, S_2, \ldots, S_N)$. The central warehouse itself has ample supply and distributes the items ordered using a homogeneous fleet with an unlimited number of vehicles, each with capacity $Q$.

We consolidate all $N$ retailers to $K$ ($1 \leq K \leq N$) mutually exclusive and collectively exhaustive retailer groups. Note that the number of retailer groups $K$, as well as the allocation of the retailers to retailer groups, is part of our optimization problem. For readability, we denote the set of retailer groups by $\mathcal{K} = \{1, \ldots, K\}$. By
using time-based shipment consolidation, all goods shipped to retail group \( k \in \mathcal{K} \) are dispatched periodically with a constant shipment interval \( T_k \in \mathbb{N}_{\geq 1} \), which refers to the time between two consecutive replenishments. Note that under time-based shipment consolidation, shipments leave the central warehouse independently of the vehicle’s current utilization but in fixed time intervals, which leads to a fixed delivery pattern. For all retailer groups \( K \) collectively, we denote the vector of shipment intervals by \( \mathbf{T} = (T_1, T_2, \ldots, T_K) \) and assume that the shipment interval is constant within each retailer group. This means that whenever a truck replenishes a retailer group, all retailers belonging to that group will be visited independently of the requested quantities.

The number of units that comprise a replenishment to retailer group \( k \) depends on the retailers’ orders according to their current inventory situation. Because shipments to retailer group \( k \) leave the warehouse every \( T_k \) periods, all retailers \( i \) belonging to that retailer group place orders every \( T_k \) periods just before the vehicle departs from the central warehouse, and hence \( R_i = T_k \). Thus, orders are placed by all retailers \( i \) belonging to retailer group \( k \) according to an \((T_k, S_i)\)-policy. To raise the inventory position after ordering to \( S_i \), the order quantity of each retailer belonging to group \( k \) equals the demand during the last \( T_k \) periods. Because demand is stochastic, order quantities are stochastic as well, which can lead to a situation where the capacity of a vehicle is not sufficient to transport all ordered units. In such a situation, an emergency shipment is performed by an external service provider to deliver excess units to the corresponding retailers at a cost per unit. We assume that both regular and emergency shipments arrive at the retailers immediately after departure from the warehouse, which means that we neglect transportation, order processing, and material handling times, because they are usually very short compared to the length of the shipment intervals. However, these processing times can easily be included in the model by increasing the lead time from the warehouse to the corresponding retailers.

Henceforth, the sequence of events in each period is as follows: At the beginning of every period, shipments arrive at all retailers belonging to a retailer group \( k \) that receive a replenishment according to the group-specific shipment interval \( T_k \). Following this, demand occurs at the retailers and is either satisfied from stock or backlogged. At the end of each period, retailers receiving an order in the next period place a new order based on the current inventory position, which comprises the stock-on-hand, backlogs, and all outstanding orders, and all costs are reported.

The aim is to find a cost-minimizing set of retailer groups, their associated shipment intervals, and the base-stock level at each retailer. The expected costs per period are composed of the sum of fixed and variable transportation costs, emergency shipment costs, and inventory holding costs for cycle and safety stock. Note that cycle stock is defined as the expected stock-on-hand required to satisfy the mean demand per replenishment cycle, whereas safety stock equals the expected stock-on-hand just before a replenishment arrives.

A detailed description of the mathematical formulation follows in Section 4, after an overview of the relevant literature related to our problem, which is presented in the next section.
3. Literature Overview

Our study interfaces with several classical streams of the operations research literature. In the following, we discuss related work in the fields of inventory management in single- and two-echelon inventory systems, capacitated clustering problems, and inventory routing problems. Because of the large number of excellent papers in all of these fields, the following overview does not claim completeness, but instead emphasizes the research gap our paper is closing.

3.1. Inventory Management

Shipment consolidation in a single-warehouse multiple-retailer setting with stochastic demand has been studied extensively. We focus on studies that take an integrated view of consolidation and inventory management, and we refer to Çetinkaya and Bookbinder (2003) and Mutlu et al. (2010) for an overview of studies that focus solely on consolidation policies, i.e., time-based, quantity-based, and time- and quantity-based shipment consolidation. As this paper considers a single-echelon system, in which ample supply is available at the central warehouse, we first review the existing literature in this field. Then, we briefly discuss the literature on two-echelon systems with limited instead of ample supply at the central warehouse.

In the stream of single-echelon models that integrate consolidation and inventory management, the grouping of retail stores is given, and the shipping capacity is unrestricted. Çetinkaya and Lee (2000) were the first to consider stochastic demand, i.e., a Poisson process, together with a time-based dispatching policy, for which they presented a heuristic solution approach. An exact solution procedure and an improved heuristic procedure were presented by Axsäter (2001). A less practical alternative to time-based dispatching policies is represented by quantity-based or hybrid dispatching policies, which were extensively compared by Chen et al. (2005) and Çetinkaya et al. (2006), who found that quantity-based policies are, in terms of costs, at least as good as time-based policies. Cetinkaya et al. (2008) presented optimal quantity-based dispatching policies for the case in which both order arrivals and order sizes are stochastic.

Owing to the complexity of the problem, the literature on shipment consolidation in the context of two-echelon distribution inventory systems with stochastic demand is limited. It has focused mainly on time-based dispatching policies because of their practicability, for instance, in the retail industry (for quantity-based consolidation policies, we refer to Kiesmüller and De Kok 2005). Considering time-based dispatching policies, Marklund (2011) and Stenius et al. (2016) presented exact approaches to minimize the total system-wide costs under Poisson and compound Poisson demand, respectively. Johansson et al. (2020) used the same model as Stenius et al. (2016) and developed computationally attractive heuristics solving larger problem instances.

All papers on single- and two-echelon inventory systems have in common that they consider an unlimited fleet of vehicles with unlimited capacity. Furthermore, geographically close retailers are replenished according to externally given retailer groups. Thus, these papers focus solely on determining shipment intervals $T$ and
base-stock levels $S$ at the retailers, and neglect the clustering and routing as well as the interaction between all these decisions. However, the importance and complexity of determining retailer groups have recently been recognized. For instance, Stenius et al. (2018) wrote that the “configuration of retailer groups can affect the performance of the system.”

3.2. Capacitated Clustering Problem

We previously discussed studies in the field of inventory management that consider joint optimization of shipment intervals $T$ and base-stock levels $S$ but take the grouping of retailers as exogenous. Therefore, the capacitated clustering problem (CCP) and the related literature are important because they focus on finding a clustering of a given number of items (e.g., retailers) into a given number of mutually exclusive and collectively exhaustive groups where the size of each group is restricted (e.g., by vehicle capacity). The objective is to minimize the distance between each item (e.g., retailer) and a designated group median, also known as the cluster center.

The CCP focuses on the allocation of retailers to clusters by taking into account geographic locations of retailers and capacity restrictions on the vehicles (see, e.g., Lorena and Senne 2004, Ceselli and Righini 2005, Geetha et al. 2009, Kargari and Sepehri 2012, Mai et al. 2018). Nevertheless, a location-based allocation of retailers to clusters can lead to low utilization of the capacitated vehicles, especially if demand is stochastic and differs significantly between retailers. Thus, the CCP can only be considered as a small subproblem of our joint optimization problem, but it serves as inspiration for some of the constructive heuristics we present in Section 6.

3.3. Inventory Routing Problem

The inventory routing problem (IRP) incorporates aspects from inventory management and capacitated clustering and is therefore most closely related to our problem. The basic version of the IRP considers one supplier and several geographically dispersed retailers. Each period, the supplier has to replenish stock at a subset of retailers using capacitated vehicles. The objective is to minimize the total inventory distribution cost while meeting the demand at each retailer (Coelho et al. 2014b).

For the replenishment of stock at the retailers, one distinguishes between static and dynamic allocation policies (Kumar et al. 1995). Under static allocation policies, the allocation of items on a vehicle to retailers is made for all retailers before the vehicle leaves the central warehouse. By contrast, under dynamic allocation policies, the allocation of vehicle inventory is postponed to the moment when the vehicle reaches each retailer. Because the solution procedure under dynamic allocation is very different from that for static allocation, which we consider in this paper, we do not review the literature on dynamic allocation but instead refer to, for instance, Trudeau and Dror (1992), Reiman et al. (1999), Jailllet et al. (2002), Schwarz et al. (2006), Huang and Lin (2010), and Ghiami et al. (2019) for papers considering this kind of problem.

Focusing on static allocation policies, the problem we consider in this paper differs substantially from classical IRPs in the following ways:
1. Many stochastic IRP papers consider that in each period, the planners are concerned with deciding which customers to replenish and how much to deliver. In contrast to this, we consider a planning problem with fixed routes to a subset of retailers that are replenished repeatedly in fixed time intervals. As mentioned before, fixed replenishment schemes are highly preferred over dynamic replenishment solutions in, for instance, the retail industry, so that related planning processes can be coordinated more efficient.

2. Those IRP papers focusing on fixed routes assume that demand rates are deterministic, whereas we consider stochastic demand, which makes it necessary to

   (i) optimize safety stock at each retailer, which depends on the replenishment frequency;

   (ii) deal with situations where not all ordered units fit on the capacitated vehicle by, for instance, considering emergency shipments.

3. Because of the complexity of the problem, most IRP papers derive heuristic instead of exact algorithms. By contrast, we present an exact algorithm as well as several heuristics to solve the problem.

Table 1 presents a summary of the IRP literature, i.e., an overview of the most relevant papers as well as those most closely related to our problem. In this overview, we emphasize the research gap between this paper and previous work in this extensively studied field of research. For extended literature reviews, we refer interested readers to the excellent papers by Kleywegt et al. (2002), Moin and Salhi (2007), and Coelho et al. (2014b). The column headings in Table 1 represent some key problem characteristics:

1. The time between two consecutive shipments—the shipment interval—can be either fixed or flexible over a particular planning horizon.

2. The demand per period at the retailers can be deterministic or stochastic.

3. This column states whether or not emergency shipments are being considered.

4. The next four columns refer to the decisions: Determining clusters of retailers that are always replenished jointly, determining fixed shipment frequencies instead of daily replenishment decisions, determining the routing between retailers and the warehouse, and determining inventory holding costs.

5. This paper has a special focus on the inventory component. Therefore, we identify papers in the IRP literature that optimize cycle stock and/or safety stock at the retailers.

6. Solution approaches can be exact or heuristic.

Table 1 divides the IRP literature into two groups. The first group focuses on a planning problem using fixed shipment intervals under deterministic demand. The second group considers a planning problem with flexible shipment intervals under stochastic demand. Under flexible shipment intervals, the probability of stock-outs at the retailers determines whether or not a retailer is replenished in the current period. Flexible shipment intervals are motivated by practical applications in, for instance, the distribution of gases to customers (see, e.g., Trudeau and Dror 1992, Kleywegt et al. 2004). In such a setting, the customers “do not routinely call requesting replenishment nor are there regular pre-scheduled deliveries” (Berman and Larson 2001). Therefore, no coordination with dependent planning processes is necessary, and day-to-day planning...
of deliveries is reasonable. Coelho et al. (2014a) are the only authors to considering emergency shipments in a daily planning environment by subcontracting to a carrier to be able to ship more than the limited vehicle capacity. The structure of this planning problem and the (mainly heuristic) solution approaches are very different compared with those required for fixed shipment intervals.

Focusing on fixed shipment intervals, the work by Gaur and Fisher (2004), who consider an IRP at Albert Heijn, a leading supermarket chain in the Netherlands, is most closely related to our problem. However, it still differs on two fundamental respects: First, it does not consider holding costs at the retailers in the objective function. Second, the solution to the inventory routing problem is based on deterministic, forecasted demand, although the actual demand is stochastic. This significantly simplifies the optimization problem that needs to be considered, because, for instance, no safety stock and no emergency cost optimization are required. Instead, violations of truck capacity, which can occur owing to extreme demand realizations, are
not considered in the planning model but must be handled by planners once they occur. However, we show in Section 7.4.2 that the impact of the corresponding costs on the structure of the optimal solution can be enormous, wherefore stochastic demands should be explicitly considered already in the planning process.

Table 1 reveals that there have been no studies considering fixed replenishment intervals for retailers facing stochastic customer demand. By considering stochastic demand at the retailers, we focus not only on optimizing cycle stock, but also on optimizing safety stock. Furthermore, we explicitly take into account emergency shipment costs in our mathematical model to deal with capacity violations. Considering all these aspects, we provide optimal and several heuristic solution approaches to this complex planning problem.

4. Model Formulation

In the following, we first formulate the mathematical model that can be used to solve our inventory routing problem with stochastic demand. We formulate it as a chance-constrained integer optimization problem. We then introduce an integer formulation by applying a Dantzig–Wolfe reformulation to the chance-constrained integer optimization problem.

4.1. Mathematical Model

First, let \( \mathcal{N}^0 := \mathcal{N} \cup \{0\} \), where \( \{0\} \) represents the central warehouse. For readability, we refer to all \( i \in \mathcal{N}^0 \) as retailers. Our problem is then defined on the graph \( G = (\mathcal{N}^0, \mathcal{A}) \), where \( \mathcal{A} \) is the complete edge set. We set \( K = N \) to ensure that we allow for all possible retailer groups in our model, because retailer groups are allowed to be empty. Recall the vectors with decision variables \( \mathbf{T} = (T_1, \ldots, T_K) \), denoting the shipment intervals of the \( K \) retailer groups, and \( \mathbf{S} = (S_1, \ldots, S_N) \), denoting the base-stock levels at each retailer. Furthermore, let \( \mathbf{Y} = (y_{ijk})_{(N+1) \times (N+1) \times K} \) denote binary decision variables that equal 1 if retailer \( j \in \mathcal{N}^0 \) is directly visited after retailer \( i \in \mathcal{N}^0 \) in retailer group \( k \in \mathcal{K} \), and equal 0 otherwise. Finally, for readability, we introduce \( \mathbf{X} = (x_{ik})_{N \times K} \) as binary decision variables that equal 1 if retailer group \( k \) contains retailer \( i \in \mathcal{N} \), and equal 0 otherwise. It holds that \( \sum_{j \in N^0} y_{ijk} = x_{ik} \) for all \( i \in \mathcal{N} \).

We first present the complete objective function. We then discuss the components of the objective function and the associated variables and parameters one-by-one. The objective function is formulated as

\[
TC(\mathbf{S}, \mathbf{T}, \mathbf{X}, \mathbf{Y}) = \sum_{k \in \mathcal{K}} \frac{W + w_k(\mathbf{Y})}{T_k} + \sum_{k \in \mathcal{K}} \mathbb{E} \left[ \left( \sum_{i \in \mathcal{N}} x_{ik} D_i(T_k) - Q \right)^+ \right] \frac{e}{T_k} + h \sum_{i \in \mathcal{N}} \sum_{k \in \mathcal{K}} \frac{1}{T_k} \sum_{t=1}^T x_{ik} \mathbb{E} \left[ \text{IL}^+_i(t, S_i) \right].
\]

(1)

The first term in Equation (1) corresponds to the regular shipment costs per period for each retailer group \( k \). These are composed of a fixed term \( W \) per replenishment and a variable component \( w_k(\mathbf{Y}) \), which
corresponds to the variable transportation costs required to replenish all retailers in group \( k \) with a single vehicle tour. The variable costs are defined as

\[
w_k(Y) = \sum_{i \in \mathcal{N}_0} \sum_{j \in \mathcal{N}_0} c_{ij} y_{ijk} \quad \forall k \in \mathcal{K},
\]

where \( c_{ij} \) is the travel cost between retailer \( i \) and retailer \( j \).

The second term in Equation (1) refers to the expected emergency shipment costs per period determined based on the expected number of units that exceed the vehicle’s capacity \( Q \) for retailer group \( k \). Here, \( D_i(T_k) \) refers to the demand at retailer \( i \) during \( T_k \) periods and \((\cdot)^+\) is the positive-part operator, i.e., \((u)^+ = \max\{0, u\}\). We assume that all excess items are delivered to the corresponding retailers by an external service provider for a fixed unit price \( e \). Because emergency shipments can occur only in periods where regular shipments take place, emergency shipment costs to retailer group \( k \) are divided by the group-specific shipment interval \( T_k \) to obtain the expected emergency shipment costs per period.

The third term in Equation (1) accumulates the expected holding costs per period over all retailers. Holding costs for a single retailer \( i \) belonging to retailer group \( k \) are calculated by multiplying the unit holding cost parameter \( h \) with the expected stock on hand \( E[IL_i^+(t, S_i)] \) at the end of each period \( t \) during the replenishment cycle of length \( T_k \). Note that the expected stock on hand depends on the base-stock level \( S_i \).

While minimizing the total expected costs per period \( TC \), several constraints need to hold such that we can find a feasible solution to our problem. First, the retailers’ target service levels \( \alpha^*_i \) must be satisfied:

\[
P(\text{IL}_i^+(T_k, S_i) > 0) \geq \alpha^*_i \quad \forall i \in \mathcal{N}, k \in \mathcal{K}.
\]

By including a service level constraint, we ensure that the probability for costly stock-outs leading to customer dissatisfaction does not exceed \( 1 - \alpha^*_i \).

Second, the capacity of all vehicles should be respected. We model this via the chance constraint

\[
P\left(\sum_{i \in \mathcal{N}} x_{ik} D_i(T_k) \leq Q\right) \geq \gamma \quad \forall k \in \mathcal{K}.
\]

This ensures that the probability, that the total demand for retailer group \( k \) during \( T_k \) periods is less than or equal to the vehicle’s capacity \( Q \), is at least \( \gamma \). In this way, we ensure that emergency shipments are the exception rather than the rule.

Summarizing, the chance-constrained nonlinear optimization model that minimizes the total expected costs per period, as defined in Equation (1), is given by

\[
\begin{align*}
\min \quad & TC(S, T, X, Y) \\
\text{s.t.} \quad & P(\text{IL}_i^+(T_k, S_i) > 0) \geq \alpha^*_i \quad \forall i \in \mathcal{N}, k \in \mathcal{K}, \\
& P\left(\sum_{i \in \mathcal{N}} x_{ik} D_i(T_k) \leq Q\right) \geq \gamma \quad \forall k \in \mathcal{K},
\end{align*}
\]
In this formulation, the constraints (6) and (7) are the chance constraints discussed previously, the constraints (8) link the $x_{ik}$ variables to the $y_{ijk}$ variables, the constraints (9) and (10) ensure that each retailer is visited exactly once, the constraints (11) enforce the condition that there is only a single tour among all retailers belonging to a single retailer group $k$, and the constraints (12)–(15) indicate the domain of the decision variables.

### 4.2. An Integer Linear Reformulation

In the remainder of this paper, we will work with an integer reformulation of the model (5)–(15). The resulting formulation is a set-partitioning model, where we need to select retailer clusters that together partition the set of retailers $\mathcal{N}$. Each cluster comprises a set of retailers with associated vehicle route, shipment interval, and corresponding individual base-stock levels in order to meet the service levels $\alpha_i^*$. Furthermore, for each cluster, it is ensured that with probability $\gamma^*$ the truck capacity is sufficiently large, as modeled via the chance constraints (7).

The formulation comprises an exponentially large set of clusters, for which column generation can be used to solve its linear relaxation. A requirement for this procedure is that the pricing problem of the set-partitioning formulation is solved. For our problem, in particular because of the chance constraints (6) and (7), this pricing problem is a new variant of the elementary resource-constrained shortest-path problem (ERCSPP). In the following, we present this integer formulation and its associated pricing problem.

Let $\mathcal{R}$ be a collection of retail clusters, where each cluster $r \in \mathcal{R}$ describes a single vehicle route along all retail stores in the cluster. Let $\beta_i^r$ be equal to 1 if retailer $i$ is contained in cluster $r$, and 0 otherwise. Furthermore, we define $T_r$ as the corresponding optimal shipment interval and $S_r^i$ as the optimal base-stock level of retailer $i$ with $\beta_i^r = 1$ (see Section 5.1.1 for their calculation).

The costs $c_r$ of cluster $r \in \mathcal{R}$ are defined as

$$c_r = \frac{1}{T_r} (w_r + W) + \frac{e}{T_r} \mathbb{E} \left[ \left( \sum_{i=1}^{N} \beta_i^r D_i(T_r) - Q \right)^+ \right].$$
\[ + h \sum_{i=1}^{N} \frac{1}{T_i} \sum_{r=1}^{R_i} \mathbb{E}[\mathcal{L}_{i}^{+}(t,S_{i})], \]

where \( w_r \) is equal to the variable transportation costs within cluster \( r \). Let \( z_r \) be binary decision variables equaling 1 if cluster \( r \) is selected and 0 otherwise. Then, the inventory routing problem with stochastic demand can be formulated as

\[
\text{MP}(z) = \min \sum_{r \in \mathcal{R}} c_r z_r
\]

s.t. \[
\sum_{r \in \mathcal{R}} \beta^r_i z_r = 1 \quad \forall i \in \mathcal{N},
\]

\[
z_r \in \{0, 1\} \quad \forall r \in \mathcal{R}.
\]

Here, the objective (17) minimizes the costs of the selected retail clusters. The constraints (18) ensure that each customer is assigned to exactly one retail cluster, and the constraints (19) indicate the domain of the variables.

In the above formulation, the set of clusters \( \mathcal{R} \) is of exponentially large size and cannot be enumerated. Instead, we consider the above master formulation restricted to a subset \( \mathcal{R} \subset \mathcal{R} \) of the variables. We call this the restricted master problem (RMP) and denote its solution value by \( \text{RMP}(z) \). The linear relaxation of the RMP [i.e., the replacement of (19) by \( z_r \geq 0 \forall r \in \mathcal{R} \)] is solved to optimality using column generation (see, e.g., Barnhart et al. 1998, Lübbecke and Desrosiers 2005).

In column generation, one iteratively generates retail clusters \( r \in \mathcal{R} \setminus \mathcal{R} \) of negative reduced cost, adds these clusters to the RMP, and consequently (re)solves the linear relaxation of the RMP. If there are no retail clusters of negative reduced cost, then the linear relaxation of the RMP is provably optimal for the linear relaxation of the MP. The major challenge in column generation is therefore to determine if there are no retail clusters of negative reduced cost left after solving the linear relaxation of the RMP. This problem is called the pricing problem, and formally asks for a solution of

\[
\min_{r \in \mathcal{R} \setminus \mathcal{R}} \hat{c}_r := c_r - \sum_{i \in \mathcal{N}} \beta^r_i \pi_i.
\]

Here, \( \hat{c}_r \) is called the negative reduced cost of cluster \( r \), and \( \pi_i \in \mathbb{R} \) are the dual variables corresponding to the constraints (18). If the pricing problem returns a cluster \( r \) with \( \hat{c}_r > 0 \), then we are certain that there are no retail clusters of negative reduced cost left, and the linear relaxation of the RMP is optimal for the linear relaxation of the MP.

Integer optimality is then obtained by embedding column generation in a branch-and-bound scheme (called branch-and-price), and its computational efficiency is enhanced by including valid inequalities. The resulting approach is called branch-price-and-cut, which we discuss in detail in Section 5.
5. Branch-Price-and-Cut Algorithm

The branch-price-and-cut algorithm to solve the formulation (17)–(19) consists of two main parts that will be discussed in the following. First (see Section 5.1), we describe a tailored labeling algorithm to solve the pricing problem (20). We detail structural insights into our problem and present a novel bounding method based on 0,1-knapsack relaxations. Second (see Section 5.2), we discuss the valid inequalities used, branching decisions, and node selection rules.

5.1. Solving the Pricing Problem

The pricing problem differs from traditional ERCSPPs in its simultaneous determination of truck routes and associated shipment intervals. The shipment interval is chosen so that it balances expected inventory holding costs, expected emergency shipment costs, and regular routing costs. Furthermore, demand is stochastic and truck capacity needs to be sufficient with probability $\gamma^\star$.

We propose a tailored labeling algorithm to solve our pricing problem. A labeling algorithm is a dynamic-programming-based method in which we iteratively extend partial retailer clusters, called labels, with a new retailer and check its feasibility. The core of its efficiency lies in the ability to prune labels based on dominance criteria. These criteria specify the conditions under which a label dominates another label, meaning that the dominated label does not need to be considered, because it will not lead to an improved solution. As well as dominance criteria, pruning based on optimality criteria is gaining popularity. Optimality criteria consider upper bounds on the possible gain that can be achieved by extending a label. We consider both dominance and optimality criteria, as we will detail in Section 5.1.2. Before that, we first define the elements of a label and the resource extension functions (REFs) for its elements in Section 5.1.1. Taking all these elements into account, the labeling algorithm solves our pricing problem by identifying retailer clusters of negative reduced cost for a given linear relaxation of the RMP.

In the remainder of this section, we assume that customer demand $D_i$ per period at retailer $i$ is gamma-distributed with retailer-dependent shape parameter $\kappa_i$ and equal scale parameter $\theta$. The advantage of considering a gamma distribution is that the probability density function can depict very different shapes and thereby cover a broad variety of possible demand distributions. Although we use a gamma distribution to model demand at the retailers, our algorithm can be adapted easily for other demand distributions (e.g., normal or Poisson distributions).

5.1.1. Label Definition, Feasibility, and Extension. To track feasibility and to apply dominance criteria, we define a label $L$ by the following elements:

1. the current partial path of retail stores $\vec{v}(L)$, with $i(L)$ denoting the retail store at the end of $\vec{v}(L)$;
2. a sorted set of unreachable retailers $U(L) \subseteq \mathcal{N}$ denoting the retailers that can no longer be visited or that are visited already;
3. the sum $\kappa(L)$ of shape parameters $\kappa_i$ of all retailers $i$ contained in $L$;
4. the incurred dual costs \( \hat{c}(L) \);
5. the total transportation costs \( w(L) \), including variable and fixed transportation costs;
6. the optimal shipment interval \( t(L) \);
7. the truck capacity used \( q(L) \), defined as the \( \gamma \% \) percentile of the joint demand distribution associated with shipment interval \( t(L) \) of all the retailers in label \( L \);
8. the objective value \( c(L) \) composed of regular and expected emergency transportation and holding costs.

The basic action of a labeling algorithm is to extend a label \( L \) with a new retailer \( i \in \mathcal{N}^0 \) into an extended label \( L' \). Before we detail the resource extension functions that update the elements of \( L \) upon extension, we explain how the optimal shipment interval \( t(L) \) can be computed for a given cluster. For better comprehensibility, we first summarize some properties of the gamma distribution. After this, we detail some properties of retailers included in a label, i.e., how to determine the base-stock level and the expected cycle and safety stock costs. Then, we indicate how to calculate the expected emergency shipment cost and how truck capacity should be respected. We present this in a sequence of three lemmas.

**Lemma 1 (Gamma distribution).** Consider a given label \( L \). For readability, assume that \( \mathcal{U}(L) \) equals the set of retailers contained in \( \mathcal{V}(L) \). Let \( t(L) \) be the associated shipment interval. Assume that each retailer \( i \in \mathcal{U}(L) \) faces gamma-distributed demand \( \Gamma(\kappa_i, \theta) \) with shape parameters \( \kappa_i \) and equal-scale parameter \( \theta \).

- The total demand at retailer \( i \) during a replenishment cycle of length \( t(L) \), \( D_i(t(L)) \), is gamma-distributed with shape parameter \( t(L)\kappa_i \) and scale parameter \( \theta \): \( D_i(t(L)) \sim \Gamma(t(L)\kappa_i, \theta) \).
- The total demand for label \( L \) per replenishment cycle of length \( t(L) \), \( D_L(t(L)) \), is also gamma-distributed with parameters \( t(L)\kappa(L) \) and \( \theta \): \( D_L(t(L)) \sim \Gamma(t(L)\kappa(L), \theta) \).

Using this lemma on the properties of the demand distribution faced by a label \( L \), we can calculate the base-stock levels at the contained retailers and determine the expected cycle and safety stock as shown in Lemma 2.

**Lemma 2 (Retailer properties).** For any given label \( L \), the following statements are true for a retailer \( i \in \mathcal{N} \) contained in \( L \):

- The optimal base-stock level at retailer \( i \) as a function of \( t(L) \) equals \( S_i(t(L)) = F^{-1}(\alpha_i^*; \kappa_i; t(L), \theta) \), where \( F^{-1}(x; a_1, a_2) \) refers to the inverse cumulative distribution function of a gamma-distributed random variable with shape parameter \( a_1 \) and scale parameter \( a_2 \) at point \( x \). The formula follows directly from the constraint (6). This result is general for any inverse cumulative distribution function \( F^{-1} \).
- The cycle stock per period under a replenishment cycle of length \( t(L) \) at retailer \( i \) equals \( \bar{d}_i(t(L)) = \frac{1}{2}\bar{d}_i(t(L)) \), where \( \bar{d}_i(t(L)) \) equals the mean demand at retailer \( i \) during \( t(L) \) periods, calculated as \( \bar{d}_i(t(L)) = t(L)\kappa_i\theta \). Note that this result is independent of the considered demand distribution.
As well as the properties of the retailers, we need to determine the expected emergency costs of label $L$. This term can be simplified under gamma-distributed demand to

$$S_i(t(L)) - d_i(t(L)) + \tilde{d}_i(t(L))(1 - F(S_i(t(L)); t(L)\kappa_i + 1, \theta)) - S_i(t(L))(1 - F(S_i(t(L)); t(L)\kappa_i, \theta)),$$

where $F(x; a_1, a_2)$ denotes the cumulative distribution function of a gamma-distributed random variable with shape parameter $a_1$ and shape parameter $a_2$ at point $x$. Note that the calculation of the safety stock as $I^t_1(t(L)) = \mathbb{E}[(S_i(t(L)) - D_i(t(L)))^+]$ is independent of the considered demand distribution. However, the simplification denoted above depends on the used demand distribution.

As well as the properties of the retailers, we need to determine the expected emergency costs of label $L$ and how the truck capacity is respected for each label. We summarize this in the following lemma:

**Lemma 3 (Label properties).** For each label $L$, the following holds:

- The expected emergency costs for label $L$ are calculated by multiplying the unit emergency cost with the expected amount of units exceeding truck capacity on the occurrence of a replenishment, given by

$$E_L(t(L)) = \mathbb{E}[(D_L(t(L)) - Q)^+]$$

$$= \int_Q^{\infty} (u - Q) f(u; t(L)\kappa(L), \theta) \, du,$$

where $f(x; a_1, a_2)$ is defined as the probability density function of a gamma-distributed random variable with shape parameter $a_1$ and scale parameter $a_2$. The integral in this equation is called the first-order loss function and can be simplified according to Silver et al. (1998) to

$$\tilde{d}_L(1 - F(Q; t(L)\kappa + 1, \theta)) - Q(1 - F(Q; t(L)\kappa(L), \theta)).$$

where $\tilde{d}_L(t(L)) = t(L)\kappa(L)\theta$ is defined as the mean demand per replenishment cycle of label $L$.

- The probability of exceeding capacity $Q$ in a replenishment cycle with shipment interval $t(L)$, $P(D_L(t(L)) > Q)$, equals $1 - F^{-1}(\gamma^*; t(L)\kappa(L), \theta)$.

Based on Lemma 2 and 3, we can calculate the optimal shipment interval $t(L)$ of label $L$ given gamma-distributed demand at each retailer, as shown in the following corollary:

**Corollary 1.** For a given label $L$, the optimal shipment interval $t(L)$ under gamma-distributed demand is given by

$$t(L) := \arg\min_{w \in \mathbb{N}^+} \{\Phi(u, L) \mid P(D_L(u) \leq Q) \geq \gamma^*\},$$

$$\Phi(u, L) = \frac{w(L)}{u} + \frac{e}{u} E_L(u) + h \sum_{i \in \mathbb{P}(L)} [I^t_1(u) + I^t_2(u)].$$
Note that $I^a_i(u)$ and $I^a_i(u)$ refer, according to Lemma 2, to the optimal expected cycle and safety stock for any given shipment interval $u$.

Having defined how the different cost components can be calculated for gamma-distributed retailer demand, we now define the resource extension functions for extending a label $L$ with retailer $j \in \mathcal{N}^0$ into label $L'$ as follows:

1. $\bar{v}(L') = (\bar{v}(L), j)$;
2. $\mathcal{U}(L') = \mathcal{U}(L) \cup \{j\}$;
3. $k(L') = k(L) + k_j$;
4. $\hat{c}(L') = \hat{c}(L) + \pi_j$;
5. $w(L') = w(L) + w_c(L, j)$;
6. $t(L') = \arg \min_{u \in \mathbb{N}^+} \left[ \Phi(u, L') \right. \text{ s.t. } P(D_{L'}(u) \leq Q) \geq \gamma';$
7. $q(L') = F^{-1}(\gamma'; t(L') k(L'); \theta)$;
8. $c(L') = \min_{u \in \mathbb{N}^+} \left\{ \Phi(u, L') \mid P(D_{L'}(u) \leq Q) \geq \gamma' \right\} - \hat{c}(L')$.

Note that in the case $j = 0$ and $\bar{v}(L')$ is an empty path, we only update the path $\bar{v}(L')$ and leave the other label elements untouched. Feasibility of a label follows trivially from the label definition; i.e., extending label $L$ with a retailer $j$ into label $L'$ is feasible if the truck capacity constraints are respected [$q(L') \leq Q$] and if $j \notin \mathcal{U}(L)$. When a label is extended with the central warehouse, i.e., if $j = 0$, we check whether $c(L') < 0$ and transform the label with negative reduced cost into a new retailer cluster and add it to the set $\tilde{\mathcal{R}}$.

5.1.2. Dominance and Optimality Criteria. Here, we introduce dominance and optimality criteria in order to prune labels during the labeling algorithm. This is crucial for the efficiency of the algorithm. If these criteria are not included, the labeling algorithm simply enumerates all possible clusters. The dominance and optimality criteria are valid under the assumption that we fix the shipment interval during execution of the labeling algorithm. Let $t^{ab} \leq t(L) \leq t^{ab}$ for all labels $L$. In the remainder of this section, we assume that the shipment interval is fixed (i.e., $t^{ab} = t^{ab}$). In Section 5.1.3, we indicate how we ensure optimality of the complete labeling algorithm.

We now discuss how a label $L$ in the labeling algorithm can be disregarded. First, we consider dominance criteria that disregard a label because there provably exists another label $L'$ that results in the same label extensions as $L$ but at lower cost. The dominance criteria are similar to those of the capacitated vehicle routing problem (see, e.g., Costa et al. 2019). For completeness, we restate these dominance criteria. A label $L$ is said to be dominated by label $L'$ if the following three conditions hold:

$$\mathcal{U}(L') \subseteq \mathcal{U}(L), \quad c(L') \leq c(L), \quad q(L') \leq q(L). \quad (23)$$

From left to right, these state that $L'$ dominates $L$ if $L'$ contains a subset of retailers only, has lower cost, and less vehicle capacity used.
Second, we provide a completion bound (i.e., an upper bound on the maximum improvement) on any possible extension of \( L \), and disregard \( L \) if it cannot result in an optimal solution to the pricing problem. The completion bound exploits the fact that the dual costs are only incurred at the retailers (and not on the actual arcs chosen) and that holding costs can be calculated directly for a fixed shipment interval. Consequently, our completion bound considers all retailers \( i \notin \mathcal{U}(L) \) and “collects” the sum of positive dual costs and the minimum holding costs at \( i \) associated with the shipment interval of label \( L \). We do this so that it results in an upper bound on the possible decrease in objective value \( c(L) \).

To obtain a valid completion bound, we need an easy-to-compute lower bound on the increase of \( q(L) \), independent of the actual demand distributions of retailers contained in \( L \). Our lower bound uses the following property of gamma-distributed demand:

**Lemma 4 (Gamma-distribution quantile).** Let \( D_i \sim \Gamma(\kappa_i, \theta) \) and \( D_j \sim \Gamma(\kappa_j, \theta) \) be gamma-distributed independent random variables, representing the demand per period of retail stores \( i \) and \( j \). Then, for any \( i, j \in \mathbb{N}, \kappa_i, \kappa_j \in \mathbb{R}^+, \theta \in \mathbb{R}^+ \) and for any parameter constellation \((\kappa_i, \kappa_j, \theta)\), there exists an \( \tilde{a}(\kappa_i, \kappa_j, \theta) \) such that for all \( a > \tilde{a}(\kappa_i, \kappa_j, \theta) \), the following inequality holds:

\[
F^{-1}(a; \kappa_i + \kappa_j, \theta) \geq F^{-1}(a; \kappa_i, \theta) + \kappa_j \theta.
\] (24)

The proof is provided in Appendix B. Lemma 4 provides a lower bound on the right tail quantile of the sum of two gamma-distributed random variables, which is useful for constructing our completion bound because it allows us to work with lower bounds on the increase of \( q(L) \) by extensions of some label \( L \). In particular, it implies that \( q(L) \) increases by less than \( \kappa_j \theta \) if some label \( L \) is extended with retailer \( j \in \mathcal{N} \).

We define for each retailer \( i \) its so-called completion costs \( CC_i \):

\[
CC_i = \frac{w}{t_{lb}} \min_{j \in S_0} c_{ij} - \pi_i + h[I_{cs,i}(t_{lb}) + I_{ss,i}(t_{lb})].
\] (25)

The completion costs \( CC_i \) consist of the shortest outgoing arc from \( i \), its associated dual costs \( \pi_i \), and the holding cost at node \( i \) with shipment interval \( t_{lb} \). For any label \( L \) and retailer \( i \notin \mathcal{U}(L) \), the completion costs \( CC_i \) are an upper bound on the decrease in \( c(L) \), because \( CC_i \) does not account for the emergency shipment costs and the shipment interval is fixed to \( t_{lb} \). In other words, it holds that the extended label \( L \cup \{i\} := L' \) has costs \( c(L') \geq c(L) + CC_i \).

The completion bound, defined for any label \( L \), then consists of selecting the most “profitable” retailers \( i \notin \mathcal{U}(L) \) according to their completion costs \( CC_i \). Using Lemma 4, we define the completion bound as the solution to the linear optimization problem

\[
\bar{z}(L) = \min_{\mathcal{Y}_i} CC_i y_i \quad \text{s.t.} \quad \sum_{i \notin \mathcal{U}(L)} \theta \kappa_i y_i \leq Q - q(L),
\] (26) (27)
0 \leq y_i \leq 1 \quad \forall i \notin \mathcal{U}(L). \quad (28)

This is a linear knapsack problem (\(\text{CC}_i \in \mathbb{R}\)), which is solved by sorting all retailers \(i \notin \mathcal{U}(L)\) in non-increasing order according to \(\text{CC}_i/(\theta \kappa)\) and assigning the \(Q - q(L)\) remaining capacity according to the sorted list of retailers. This sorting operation is done before the labeling algorithm starts and does not add complexity to solving the pricing problem. However, the assignment of remaining capacity \(Q - q(L)\) to retailers \(i \notin \mathcal{U}(L)\) depends on \(L\) and is performed when the optimality pruning criteria are invoked. We summarize these optimality pruning criteria by our completion bound in the following lemma:

**Lemma 5 (Completion bound).** Let \(L\) be a given label, and let \(\bar{z}(L)\) be defined as above. Let \(\bar{z}^0\) be the best solution found so far during the execution of the labeling algorithm. If \(c(L) + \bar{z}(L) > \bar{z}^0\), then any extension of label \(L\) will result in a label of cost larger than \(\bar{z}^0\), and label \(L\) can be pruned.

5.1.3. Labeling Algorithm Procedure. The labeling algorithm procedure works as follows. After observing the LP relaxation to the RMP, we iteratively run the labeling algorithm for the parameter \(\bar{t} \in \{\bar{t}^{\max}, \bar{t}^{\max} - 1, \ldots, 1\}\). For \(\bar{t} = \bar{t}^{\max}\), we set \(t^{\text{ub}} = \infty\) and \(t^{\text{lb}} = \bar{t}\). For other values of \(\bar{t}\), we impose \(t^{\text{ub}} = t^{\text{lb}} = \bar{t}\). Note that in the case \(t^{\text{ub}} = \infty\), our optimality pruning criteria are not valid, which is not a problem, because setting \(\bar{t}^{\max}\) sufficiently large ensures that the vehicle capacity is very restricted in this case. In our experiments, we set \(\bar{t}^{\max} = 4\). We abort the enumeration over \(\bar{t}\) if for some \(\bar{t}\) we identified retailer clusters of negative reduced cost. We then solve the linear relaxation of the RMP and restart our iterative labeling algorithm. We terminate a single call to the labeling algorithm procedure after we have found 5000 clusters of negative reduced cost, after which we resolve the RMP and restart the labeling algorithm procedure.

5.2. Branching and Valid Inequalities

Valid branching rules are required to obtain a working exact solution method. We make use of two branching rules. First, we branch on an integral number of vehicles being used. For a given LP relaxation to the RMP, we define \(z^*_j := \sum_{r \in \mathcal{R}} \delta^*_j z^*_r\), where \(\delta^*_j\) is a binary parameter indicating if retailer \(j\) is visited directly after retailer \(i\) in cluster \(r\), and \(z^*_r\) is the value of \(z^*_r\) in the LP relaxation of the RMP. Then, if \(\sum_{j \in \mathcal{N}} z^*_0j\) is fractional, we create two child nodes where we impose \(\sum_{r \in \mathcal{R}} \delta^*_0j z^*_r \leq \left\lfloor \sum_{j \in \mathcal{N}} z^*_0j \right\rfloor\) and \(\sum_{r \in \mathcal{R}} \delta^*_0j z^*_r \geq \left\lceil \sum_{j \in \mathcal{N}} z^*_0j \right\rceil\), respectively. If an integral number of vehicles is used, we continue with branching on individual arcs. That is, we select the arc \((i, j)\) for which \(z^*_ij\) is closest to 0.5, and create two child nodes: one child node where we enforce arc \((i, j)\) to be traversed by at least a single cluster, and one in which we do not allow arc \((i, j)\) to be traversed. Note that the branching rule on the number of vehicles imposes cuts on the model formulation, which we take into account in our pricing problem by initializing the dual cost component of our labeling algorithm with the dual values associated with the branching cut. Furthermore, the branch rule on individual arcs requires to dynamically adjust the set of generated clusters during the branch-and-bound search by setting local upper bounds of 0.0 in the corresponding master variables. Node selection is done using the default node selection
method from the constrained programming environment SCIP 6.0.2 (Gleixner et al. 2018), which we use to code our branch-price-and-cut algorithm.

In addition, we add subset-row inequalities (Jepsen et al. 2008) to further strengthen our LP relaxation. Subset-row inequalities are defined for an $\mathcal{O} \subset \mathcal{N}$ of retailer nodes and an integer $1 \leq \phi \leq |\mathcal{O}|-1$, and are given by

$$\sum_{r \in \hat{\mathcal{R}}} \left( \frac{1}{\phi} \sum_{i \in \mathcal{O}} \beta^i_r \right) z_r \leq \left\lfloor \frac{|\mathcal{O}|}{\phi} \right\rfloor.$$

(29)

Preliminary experiments have shown that only subset-row inequalities considering $|\mathcal{O}| = 3$ and $\phi = 2$ have a significant effect on the quality of the root node relaxation. We have therefore only included those in our branch-price-and-cut algorithm. Separation is done via complete enumeration and is considered in every node of the branch-and-bound tree, because it requires negligible time compared with solving the pricing problems. Finally, including these inequalities requires some standard adaptations of our labeling algorithm, because the subset-row inequalities considered in this paper are so-called “non-robust cuts.” We have made similar changes to those outlined in, for instance, Costa et al. (2019), Schrotenboer et al. (2019).

6. Heuristics

In this section we introduce three newly developed constructive heuristics inspired by classical heuristics from the capacitated clustering and vehicle routing literature, and a MIP-based heuristic. All the four heuristics serve as upper bounding methods in our branch-price-and-cut algorithm, and as constructive heuristics in general. We introduce all heuristics briefly here and then provide further details in the remainder of this section.

**KM:** The first upper bounding method is a tailored version of a $K$-means algorithm, which is the “best-known clustering algorithm” (Geetha et al. 2009) and is commonly used to solve the capacitated clustering problem (see Section 3.2).

**SAV:** The second heuristic is based on ideas from the classical savings algorithm by Clarke and Wright (1964) for the vehicle routing problem (VRP).

**KMSAV:** The third heuristic is a combination of the first two algorithms. It uses ideas inspired by $K$-means to cluster retailers, and applies the savings-based heuristic within the generated clusters.

**MIP-H:** The fourth method solves the set-partitioning formulation RMP without using column generation. We populate the route set with all the clusters that are encountered while executing the three constructive heuristics introduced above. Then, we rely on SCIP to solve to MIP to optimality.

These heuristics serve two purposes. First, they provide good upper bounds very quickly, which is of practical relevance. Second, the clusters generated by the heuristics will be used as an initial set of clusters for the branch-price-and-cut algorithm. This speeds up the overall run-time of the branch-price-and-cut algorithm by potentially reducing the number of columns to be generated and by providing a better starting point for primal heuristics included by default in SCIP.

Our $K$-means algorithm is outlined in Algorithms 1 and 2. The $K$-means heuristic requires the maximum number of clusters $K$ as an input. To handle this, we condition our analysis on this number, which can range from 1 to $N$. We also condition our analysis on the minimum shipment interval $t$, which can range from 1 to the maximum shipment interval under direct deliveries. Enumerating over imposed minimum shipment intervals broadens the search, because it prevents large clusters from being formed. After a clustering into $k$ retailer groups has been obtained for a minimum shipment interval $t$, we optimize its routing, shipment interval, and base-stock levels via the function “getCosts” (see Section 5.1.1 for the calculations).

Algorithm 1: (KM): $K$-means heuristic

\[
\begin{align*}
&c^* \leftarrow \infty; \\
&P^* \leftarrow \emptyset; \\
&\text{for } \text{useRegret} \in \{\text{true}, \text{false}\} \text{ do} \\
&\quad \text{for } \text{useCW} \in \{\text{true}, \text{false}\} \text{ do} \\
&\quad \quad \text{for } t \in \{1, 2, 3, \ldots, 0.5N\} \text{ do} \\
&\quad \quad \quad \text{for } k \in \{2, 3, \ldots, N\} \text{ do} \\
&\quad \quad \quad \quad P \leftarrow \text{findAssignmentToKMeans}(t, k, \text{useRegret}, \text{useCentralWarehouse}); \\
&\quad \quad \quad \quad c \leftarrow \text{getCosts}(P); \\
&\quad \quad \quad \quad \text{if } (c < c^*) \text{ then} \\
&\quad \quad \quad \quad \quad c^* \leftarrow c; \\
&\quad \quad \quad \quad \quad P^* \leftarrow P; \\
&\quad \quad \quad \quad \text{end} \\
&\quad \quad \text{end} \\
&\quad \text{end} \\
&\text{return } P^*
\end{align*}
\]

Given the number of clusters and a minimum shipment interval, we iteratively assign a sorted list of retailers to clusters via the function “findAssignmentToKMeans”. The Boolean argument “useRegret” indicates whether or not we sort retailers based on the expected demand per period or if we sort the retailers based on a regret function. The Boolean argument “useCW” indicates whether or not we include the location of the central warehouse in our calculation of the clusters’ centroids. All retailers are assigned in the respective order to their nearest feasible cluster. Here, “nearest” is defined as the distance to the centroid of the cluster, which is calculated by averaging the coordinates of all included retailers. We initialize the centroids with randomly selected retailer coordinates. It is feasible to incorporate an additional retailer into a cluster if sufficient vehicle capacity remains to fit the retailer’s demand during $t$ periods. After all retailers have been assigned, we update the centroids of the $k$ clusters, based on the assigned retailers. We define the so-called deviation as the sum of distances between the old and updated clusters’ centroids. If this deviation is smaller
Algorithm 2: findAssignmentToKMeans($t, k, useRegret, useCentralWarehouse$)

$c^* \leftarrow \infty$;

$P^* \leftarrow$ initializeCentroids($k$);

for $i \in \{1, \ldots, i_{\text{max}}\}$ do

R $\leftarrow$ getSortedRetailersArray(useRegret);

for ($r \in R$) do

assignRetailerToClosestFeasibleCentroid($r, P$)

end

$d \leftarrow$ updateCentroidsAndReturnDeviation($P$, useCentralWarehouse);

if ($d < \varepsilon$) then

return $P$

end

end

return $P$

than $\varepsilon$, then we return the current clustering. Otherwise, we remove all customers and start the iterative
process over again by using the coordinates of the centroids of the previous iteration as initialization. Because
this process might not converge, we impose a maximum of $i_{\text{max}}$ iterations (see Algorithm 2) and select the
best solution found.

6.2. Savings Heuristic (SAV).

Our savings algorithm is outlined in Algorithms 3 and 4.

Algorithm 3: (SAV): Savings heuristics

$P = \text{initial set of clusters}$;

while true do

$p_1, p_2 \leftarrow$ findClustersToMerge($P$);

if ($p_1 == \emptyset || p_2 == \emptyset$) then

break;

end

$p \leftarrow$ merge($p_1, p_2$);

$P$.remove($p_1, p_2$);

$P$.add($p$);

end

return $P$

The algorithm starts by initializing the set $P$ with clusters consisting of single retailers that are re-
plenished using direct deliveries as in Kleywegt et al. (2002). Then, the algorithm iteratively attempts to
merge two clusters that result in the highest cost decrease. Each time we evaluate merging two clusters,
we calculate the exact routing costs of the new cluster (via the function “getTravelCosts”) by calling a
Algorithm 4: findClustersToMerge($P$)

\[ \begin{align*}
    c^* & \leftarrow \infty; \\
    p_1^*, p_2^* & \leftarrow \emptyset; \\
    \text{for } p_1, p_2 \in P \text{ do} & \\
    \quad c & \leftarrow \text{getTravelCosts}(p_1, p_2); \\
    \quad c & \leftarrow c + \text{getHoldingandEmergencyCosts}(p_1, p_2); \\
    \quad \text{if } (c - c(p_1) - c(p_2) < c^*) & \text{ then} \\
    \quad \quad c^* & \leftarrow c; \\
    \quad \quad p_1^* & \leftarrow p_1; \\
    \quad \quad p_2^* & \leftarrow p_2; \\
    \text{end} & \\
\end{align*} \]

return $p_1^*, p_2^*$;

straightforward mixed-integer programming (MIP) implementation of the traveling salesman problem. Given the exact routing costs, we then determine optimal shipment intervals and base-stock levels via the function “getHoldingAndEmergencyCosts”. This procedure is repeated until there are no longer any clusters whose merger would lead to a cost reduction. We impose a maximum time of 10 s for the TSP to be solved in the evaluation of merging two clusters. In the rare case in which the TSP is not solved to optimality within this time limit, we take the best incumbent solution. During our experiments, this time limit was almost never reached, and increasing it did not lead to significant improvements of the heuristic. To speed up the calculations, we store each of the solved TSPs in memory, and, on calling the MIP solver, we first check if we have already calculated this tour once before. Note, we rely on using a MIP solver to solve the TSP (instead of, for instance, an LKH inspired approach), so that optimality is proven and the resulting clusters can be reused for our exact method.

6.3. **K-means + Savings Algorithm (KMSAV).**

Our third constructive heuristic is a combination of the K-means algorithm and the savings algorithm. It starts with the K-means algorithm in its most classical sense, ignoring capacity restrictions. We simply assign retailers, which are sorted on their expected demand per period, to the centroid of the nearest clusters. We then apply our saving algorithm to each of the “regions” thereby created. The advantage of this two-step algorithm over K-means (KM) is that it takes into account demand quantities and therefore vehicle utilization more accurately, because final retailer groups are determined using the savings algorithm. The advantage of KMSAV over the pure savings algorithm (SAV) is that the allocation of retailers to regions prevents the savings algorithm from combining two retailers that are not geographically close in one shipment. This overcomes the disadvantage of the savings algorithm that it myopically consolidates two retailer groups, which might not lead to a good overall clustering.
6.4. MIP heuristic (MIP-H).

Our fourth heuristic builds upon the exploration of retailer clusters by the three constructive heuristics (KM, SAV, and KMSAV). During the execution of each of the three heuristics, we consider many different retailer clusters, all of which we store in memory together with their corresponding optimal shipment intervals and base-stock levels. The MIP-H heuristic then considers the set-partitioning formulation RMP subject to all considered clusters and solves the resulting formulation with this fixed set of clusters. By construction, this heuristic is at least as good as the three constructive heuristics, but it requires these three to be executed first. Nevertheless, the computation time of MIP-H excluding the computation time of the constructive heuristics is of the order of a few seconds for all instances considered in the remainder of this paper.

7. Computational Results

This section aims to evaluate the performance of the branch-price-and-cut method and the heuristics described in Section 6. As well as the heuristics SAV, KM, KMSAV, and MIP-H, we also consider a heuristic variant of our branch-price-and-cut method that only considers column generation in the root node. We call this the MIP-CG heuristic. We abbreviate our complete branch-price-and-cut algorithm as MIP-BPC.

All algorithms are implemented in C++17, using the framework for constrained programming SCIP 6.0.2 and CPLEX 12.8. All the implementations are completely single-threaded. The experiments are performed on an Intel Xeon E5 2680v3 2.5 GHz CPU with 40 GB of RAM allocated.

In the following, we first introduce the benchmark instances used in Section 7.1. Then, in Section 7.2, we evaluate the performance of our exact and heuristic methods on the benchmark instances. In Section 7.3, we illustrate the importance of considering stochastic customer demand, by studying the value of the stochastic solution. In Section 7.4, we compare the structure of the optimal and heuristic solutions, and show how the different cost components steer the structure of the optimal solution.

7.1. Instance Characteristics

We create a benchmark set where the warehouse is fixed to coordinates \((50, 50)\) in a \(100 \times 100\) box. We vary the number of retailers \(N\) from 20 to 60 in steps of 5, the truck capacity \(Q\) equals either 70 or 90, the holding cost parameter \(h\) equals either 0.5 or 1.0, and the travel cost \(c_{ij}\) equals the Euclidean distance between nodes \(i\) and \(j\). The emergency shipment costs \(e\) equal 50 and the fixed vehicle cost \(W\) is set to 100. The retailers’ target service levels \(\alpha_i^*\) are set to 0.95, and truck capacity should be sufficient with probability 0.9, i.e., \(\gamma^* = 0.9\).

For each combination of these parameters, we randomly create 10 instances. The randomness thereby lies in the demand distribution and the location of retailers which is drawn uniformly from the \(100 \times 100\) box. For the demand at the retailers, we set \(\theta = 15/16\) and randomly draw \(\kappa_{i}\) for each retailer \(i\) between 10 and 22. Thus, the average shape parameter \(\bar{\kappa}\) equals 16, which corresponds to an average coefficient of variation of \(\text{CoV} = \sqrt{\bar{\kappa}\theta}/\bar{\kappa}\theta = 1/\sqrt{\bar{\kappa}} = 0.25\).
### Table 2 Performance of MIP-BPC and MIP-CG

<table>
<thead>
<tr>
<th>N</th>
<th>Q</th>
<th>h</th>
<th>#rep</th>
<th>#sol</th>
<th>Gap (%)</th>
<th>max Gap (%)</th>
<th>time&lt;sub&gt;opt&lt;/sub&gt; (s)</th>
<th>#sol</th>
<th>Diff&lt;sub&gt;UB&lt;/sub&gt; (%)</th>
<th>Diff&lt;sub&gt;LB&lt;/sub&gt; (%)</th>
<th>time (s)</th>
<th>time&lt;sub&gt;opt&lt;/sub&gt; (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>-</td>
<td>30</td>
<td>10</td>
<td>0.04</td>
<td>0.04</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>0.03</td>
<td>0.03</td>
<td>8</td>
<td>8</td>
<td></td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>0.07</td>
<td>0.07</td>
<td>16</td>
<td>16</td>
<td></td>
<td></td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>-</td>
<td>65</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>7</td>
<td>7</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>415</td>
<td>0.00</td>
<td>44</td>
<td>44</td>
<td></td>
<td></td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>-</td>
<td>2976</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>70</td>
<td>70</td>
<td></td>
<td></td>
<td>70</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>32</td>
<td>10</td>
<td>10</td>
<td>0.02</td>
<td>0.02</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>-</td>
<td>80</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>14</td>
<td>14</td>
<td></td>
<td></td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>554</td>
<td>0.01</td>
<td>108</td>
<td>108</td>
<td></td>
<td></td>
<td>108</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.22</td>
<td>917</td>
<td>10</td>
<td>0.17</td>
<td>0.21</td>
<td>151</td>
<td>151</td>
<td></td>
<td></td>
<td>151</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>69</td>
<td>10</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>27</td>
<td>27</td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>-</td>
<td>3221</td>
<td>10</td>
<td>0.04</td>
<td>0.04</td>
<td>32</td>
<td>32</td>
<td></td>
<td></td>
<td>32</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>9</td>
<td>0.10</td>
<td>1064</td>
<td>0.09</td>
<td>425</td>
<td>425</td>
<td></td>
<td></td>
<td>425</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.27</td>
<td>1953</td>
<td>10</td>
<td>0.09</td>
<td>0.20</td>
<td>550</td>
<td>550</td>
<td></td>
<td></td>
<td>550</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>246</td>
<td>10</td>
<td>10</td>
<td>0.00</td>
<td>0.00</td>
<td>44</td>
<td>44</td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.16</td>
<td>3642</td>
<td>10</td>
<td>0.11</td>
<td>0.12</td>
<td>49</td>
<td>49</td>
<td></td>
<td></td>
<td>49</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>6</td>
<td>0.05</td>
<td>5387</td>
<td>0.01</td>
<td>597</td>
<td>597</td>
<td></td>
<td></td>
<td>597</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.70</td>
<td>8214</td>
<td>10</td>
<td>0.08</td>
<td>0.26</td>
<td>865</td>
<td>865</td>
<td></td>
<td></td>
<td>865</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>10</td>
<td>-</td>
<td>502</td>
<td>10</td>
<td>10</td>
<td>0.06</td>
<td>0.06</td>
<td>90</td>
<td>90</td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.13</td>
<td>4027</td>
<td>10</td>
<td>0.08</td>
<td>0.15</td>
<td>97</td>
<td>97</td>
<td></td>
<td></td>
<td>97</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>6</td>
<td>0.44</td>
<td>6325</td>
<td>0.07</td>
<td>1653</td>
<td>1653</td>
<td></td>
<td></td>
<td>1653</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>3</td>
<td>14.70</td>
<td>4435</td>
<td>10</td>
<td>0.12</td>
<td>2056</td>
<td>2056</td>
<td></td>
<td></td>
<td>2056</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>70</td>
<td>0.5</td>
<td>9</td>
<td>0.10</td>
<td>2319</td>
<td>0.02</td>
<td>125</td>
<td>125</td>
<td></td>
<td></td>
<td>125</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.27</td>
<td>3284</td>
<td>10</td>
<td>0.11</td>
<td>0.22</td>
<td>133</td>
<td>133</td>
<td></td>
<td></td>
<td>133</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>5</td>
<td>0.33</td>
<td>8833</td>
<td>0.06</td>
<td>2102</td>
<td>2102</td>
<td></td>
<td></td>
<td>2102</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>1.05</td>
<td>5060</td>
<td>10</td>
<td>0.48</td>
<td>0.25</td>
<td>2409</td>
<td>2409</td>
<td></td>
<td></td>
<td>2409</td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>7</td>
<td>0.08</td>
<td>0.02</td>
<td>234</td>
<td>234</td>
<td></td>
<td></td>
<td>234</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.13</td>
<td>6195</td>
<td>10</td>
<td>0.09</td>
<td>0.16</td>
<td>269</td>
<td>269</td>
<td></td>
<td></td>
<td>269</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>1</td>
<td>3.79</td>
<td>16572</td>
<td>0.19</td>
<td>4730</td>
<td>4730</td>
<td></td>
<td></td>
<td>4730</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>8</td>
<td>13.10</td>
<td>6635</td>
<td>7</td>
<td>0.63</td>
<td>6376</td>
<td>5322</td>
<td>5322</td>
<td></td>
<td></td>
<td>5322</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>70</td>
<td>0.5</td>
<td>10</td>
<td>7</td>
<td>0.11</td>
<td>0.02</td>
<td>313</td>
<td>313</td>
<td></td>
<td></td>
<td>313</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>10</td>
<td>0.25</td>
<td>7194</td>
<td>10</td>
<td>0.18</td>
<td>0.10</td>
<td>306</td>
<td>306</td>
<td></td>
<td></td>
<td>306</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>10</td>
<td>0</td>
<td>2.84</td>
<td>7.31</td>
<td>-</td>
<td>2.53</td>
<td>2.53</td>
<td></td>
<td></td>
<td>2.53</td>
<td></td>
</tr>
<tr>
<td>1.0</td>
<td>8</td>
<td>0.399</td>
<td>9.83</td>
<td>-</td>
<td>8</td>
<td>-3.40</td>
<td>0.49</td>
<td>9459</td>
<td></td>
<td></td>
<td>9459</td>
<td></td>
</tr>
</tbody>
</table>

#sol: number of instances solved before time limit is reached as a fraction of number of instances that processed the root node before time or memory limit is reached. Gap (%): Average optimality gap of instances not solved to optimality. max Gap (%): Maximum optimality gap of instances not solved to optimality. Diff<sub>UB</sub> (%): Average difference to upper bound of MIP-BPC. Diff<sub>LB</sub> (%): Average difference to lower bound of MIP-BPC. time<sub>opt</sub> (s): average run time in seconds of instances solved to optimality by MIP-BPC. time (s): Average run time over all instances.

### 7.2. Performance of the Exact and Heuristic Methods

In Table 2, we provide an overview of the performance of MIP-BPC and MIP-CG on the benchmark set. For each combination of parameters, we consider 10 randomly drawn instances, but on some instances no root node could be processed by the exact method within our specified time and memory limit. The column “#inst” denotes the number of instances whose root node could be processed within our prespecified time limits. We considered only those instances for which we obtained a root node solution in our comparison of the average performance of MIP-BPC and MIP-CG in each row of Table 2. The column “#sol” indicates how many of the #inst instances are solved to optimality (for MIP-BPC) or until convergence (for MIP-CG), the column “Gap (%)” shows the average optimality gap for the instances not solved to optimality, the column “max Gap (%)” shows the maximum optimality gap, and the column “time<sub>opt</sub> (s)” shows the average run time over all instances.
computation time of the instances solved to optimality by MIP-BPC. For MIP-CG, we compare the resulting upper bound with the upper bound of MIP-BPC (the column “DiffUB”) and the lower bound of MIP-BPC (the column “DiffLB”). Note that negative values for DiffUB indicate improvements over the upper bounds found by the exact method, which can happen in case the solution is not solved to optimality. Finally, the column “time (s)” gives the average computation time on instances solved to optimality or until convergence by MIP-CG.

In Table 2, a few observations stand out. MIP-BPC can solve instances to optimality up to \( N = 60 \) retailers, although computation times increase and the performance of MIP-BPC is affected by vehicle capacity and holding costs. Higher capacity and holding costs lead to an increase in cluster sizes, and therefore to a larger solution space, which increases the computational efforts. The reported average optimality gaps are, if not solved to optimality, rather small for MIP-BPC, and also the maximum optimality gaps are within a 14.7% range. Notice that for the larger instances, solving the pricing problem in each branch-and-bound node is a rather time-consuming aspect of MIP-BPC, which may impair its ability to search for high-quality upper bounds.

In comparison, the performance of MIP-CG is outstanding. It converges on all instances except two, with the average deviation from the lower bound of MIP-BPC being only 0.11%. On the smaller instances, it typically finds the same solution as MIP-BPC, and on larger instances it exploits its relatively fast branch-and-bound process compared with MIP-BPC to achieve significantly better upper bounds, although at the expense of missing the optimal solution with a small probability. On the instances with more than 50 customers and high vehicle capacity, the upper bounds of MIP-CG outperform those of MIP-BPC by a few percentages. Comparing the computation times on the instances solved to optimality by MIP-BPC, we observe that MIP-CG only requires 251 s on average, compared with 1951 s for MIP-BPC.

In Table 3, we show the performance of the heuristics. First, we compare the MIP-H heuristic with the upper and lower bounds of MIP-BPC. We can see that, especially on larger instances, MIP-H outperforms the solution found by MIP-BPC. Although the same initial columns are used in MIP-H and MIP-BPC, the column generation procedure consumed so much time that the default solver behavior could not come up with the same upper bounds. The average difference from the lower bound equals 0.62%, which emphasizes the excellent performance of MIP-H. Because its computation time is negligible, this method is very attractive for practical applications. As well as analyzing the performance of MIP-H, Table 3 also presents analyses for the other heuristics, namely, KMSAV, SAV, and KM. We compare the upper bounds from these heuristics with the solution found by MIP-H. Table 3 shows that KMSAV outperforms SAV and KM on average. For a more detailed analysis regarding the performance of the heuristics we refer to Section 7.4.1.

Summarizing, we can solve medium size instances up to 30 retailers for almost all instances to optimality. Because the performance of MIP-BPC deteriorates with increasing problem size, MIP-CG can be used to generate near-optimal solutions for large instances in reasonable time. The hybrid heuristic MIP-H returns
Table 3 Performance of MIP-H and the Constructive Heuristics KMSAV, SAV, and KM

<table>
<thead>
<tr>
<th>N</th>
<th>Q</th>
<th>h</th>
<th>MIP-H Diff_MIP-BPC (%)</th>
<th>MIP-H Diff_MIP-BPC (%)</th>
<th>KM Diff_MIP-H (%)</th>
<th>SAV Diff_MIP-H (%)</th>
<th>KMSAV Diff_MIP-H (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>70</td>
<td>0.5</td>
<td>0.10</td>
<td>0.10</td>
<td>2.94</td>
<td>3.01</td>
<td>3.37</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0</td>
<td>0.50</td>
<td>0.50</td>
<td>5.08</td>
<td>5.94</td>
<td>3.32</td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>0.19</td>
<td>0.19</td>
<td>1.18</td>
<td>1.6</td>
<td>2.58</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.69</td>
<td>0.69</td>
<td>1.74</td>
<td>3.11</td>
<td>3.33</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>70</td>
<td>0.5</td>
<td>0.14</td>
<td>0.16</td>
<td>2.95</td>
<td>3.64</td>
<td>2.31</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.64</td>
<td>0.64</td>
<td>1.4</td>
<td>1.95</td>
<td>3.16</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>0.40</td>
<td>0.40</td>
<td>1.15</td>
<td>2.87</td>
<td>2.09</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>1.13</td>
<td>1.13</td>
<td>2.58</td>
<td>5.91</td>
<td>2.73</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>70</td>
<td>0.5</td>
<td>0.25</td>
<td>0.25</td>
<td>2.41</td>
<td>2.62</td>
<td>3.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0</td>
<td>0.85</td>
<td>0.85</td>
<td>4.71</td>
<td>5.91</td>
<td>2.73</td>
</tr>
<tr>
<td>35</td>
<td>70</td>
<td>0.5</td>
<td>0.18</td>
<td>0.18</td>
<td>2.75</td>
<td>3.14</td>
<td>4.06</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.82</td>
<td>0.82</td>
<td>4.22</td>
<td>4.64</td>
<td>2.85</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>0.5</td>
<td>0.56</td>
<td>0.57</td>
<td>1.41</td>
<td>2.08</td>
<td>3.52</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>1.34</td>
<td>1.45</td>
<td>2.00</td>
<td>3.18</td>
<td>2.14</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>70</td>
<td>0.5</td>
<td>0.30</td>
<td>0.30</td>
<td>2.96</td>
<td>3.24</td>
<td>4.12</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0</td>
<td>0.65</td>
<td>0.67</td>
<td>4.36</td>
<td>5.92</td>
<td>3.29</td>
</tr>
<tr>
<td>45</td>
<td>70</td>
<td>0.5</td>
<td>0.51</td>
<td>0.53</td>
<td>1.49</td>
<td>2.12</td>
<td>3.89</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.32</td>
<td>0.67</td>
<td>2.65</td>
<td>4.04</td>
<td>3.41</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>70</td>
<td>0.5</td>
<td>0.32</td>
<td>0.32</td>
<td>2.69</td>
<td>2.95</td>
<td>4.23</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.66</td>
<td>0.73</td>
<td>4.32</td>
<td>5.42</td>
<td>3.76</td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>70</td>
<td>0.5</td>
<td>0.30</td>
<td>0.48</td>
<td>2.24</td>
<td>2.78</td>
<td>4.17</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.0</td>
<td>-0.57</td>
<td>1.03</td>
<td>2.51</td>
<td>3.8</td>
<td>3.04</td>
</tr>
<tr>
<td>60</td>
<td>70</td>
<td>0.5</td>
<td>0.38</td>
<td>0.39</td>
<td>2.68</td>
<td>2.85</td>
<td>4.3</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.64</td>
<td>0.75</td>
<td>4.75</td>
<td>5.95</td>
<td>3.47</td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>70</td>
<td>0.5</td>
<td>0.40</td>
<td>0.57</td>
<td>2.51</td>
<td>3.55</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.07</td>
<td>0.80</td>
<td>2.98</td>
<td>4.39</td>
<td>3.43</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>70</td>
<td>0.5</td>
<td>0.29</td>
<td>0.32</td>
<td>3.17</td>
<td>3.58</td>
<td>4.45</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.51</td>
<td>0.59</td>
<td>5.02</td>
<td>6.09</td>
<td>3.46</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>0.5</td>
<td>-2.50</td>
<td>0.74</td>
<td>2.27</td>
<td>2.86</td>
<td>4.07</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>-7.10</td>
<td>1.12</td>
<td>3.15</td>
<td>4.01</td>
<td>3.18</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>0.5</td>
<td>0.40</td>
<td>0.43</td>
<td>3.24</td>
<td>3.50</td>
<td>4.49</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>0.50</td>
<td>0.67</td>
<td>5.89</td>
<td>6.93</td>
<td>3.58</td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>0.5</td>
<td>-2.02</td>
<td>0.74</td>
<td>2.04</td>
<td>2.56</td>
<td>3.82</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>-2.73</td>
<td>1.18</td>
<td>2.68</td>
<td>3.32</td>
<td>2.96</td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>0.01</td>
<td>0.62</td>
<td>2.86</td>
<td>3.65</td>
<td>3.42</td>
</tr>
</tbody>
</table>

Diff_MIP-BPC (%) (average difference to upper bound of MIP-BPC). Diff_MIP-BPC (%) (average difference to lower bound of MIP-BPC). Diff_MIP-H (%) (average difference to MIP-H).

Excellent solutions in negligible time, and is therefore an attractive method solving practically sized problem instances.

7.3. The Value of the Stochastic Solution

To study the impact of demand uncertainty on the composition of retailer groups and costs, we provide a deterministic counterpart to the solution of MIP-CG. This so-called expected value solution is obtained by considering deterministic retailer demand equal to the mean demand in our stochastic model. We set the available truck capacity to $\gamma Q$, which is equivalent to considering $(1 - \gamma)Q$ buffer space on each vehicle. Note that emergency shipments and safety stocks are not required if demand is deterministic which reduces the complexity significantly.
To overcome the unrealistically low service levels at the retailers in case one applies the expected value solution to our stochastic setting, we enhance the expected value solution by re-calculating the corresponding optimal shipment intervals and base-stock levels by considering the actual stochastic demand distributions while keeping the retailer clustering fixed. Although we re-optimize shipment intervals, truck capacity might still be insufficient, which is compensated by an increase in emergency costs.

The results are presented in Table 4 and show that the adjusted expected value (AEV) solution, where shipment intervals and base stock levels are re-optimized, is on average 7.7% worse than MIP-CG. Besides, 94.7% of the instances contain at least one cluster were the probability that truck capacity is sufficient is below $\gamma^*$, which also leads to an increase of emergency shipment costs from 3.1% to 6.9% of the total costs. Hence, accounting for uncertainty in the demand distribution in our joint optimization problem is of crucial importance to control costs.

### 7.4. Structure of the Optimal and Heuristic Solutions

We continue by providing further insights in the structure of the optimal and the heuristic solutions. In Section 7.4.1, we compare the solutions between algorithms, and in Section 7.4.2 we show the impact of the different cost components on the optimal solution.

#### 7.4.1. Comparison Between Algorithms

In Table 5, we compare the average cluster size, the average shipment interval, and the average cost composition between MIP-CG, which yields as proxy for the optimal solution, and KMSAV, SAV, and KM. The first column in Table 5 shows the average absolute cluster sizes and shipment intervals for MIP-CG. Furthermore, it shows the travel, emergency and holding costs as a percentage of the total costs. The remaining columns reflect the average percentage deviation of KMSAV, SAV and KM from MIP-CG.

All constructive heuristics lead to solutions with a higher number of clusters on average, especially for
Table 5  Characteristics of the Constructive Heuristics KMSAV, SAV, and KM Compared to MIP-CG, Averaged over all Instances of the Benchmark Set

<table>
<thead>
<tr>
<th></th>
<th>Value</th>
<th>% Diff. to MIP-CG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MIP-CG</td>
<td>KMSAV</td>
</tr>
<tr>
<td>Average number of clusters</td>
<td>11.42</td>
<td>34.3</td>
</tr>
<tr>
<td>Average shipment interval</td>
<td>1.5</td>
<td>35.9</td>
</tr>
<tr>
<td>Average total cost</td>
<td>–</td>
<td>3.6</td>
</tr>
<tr>
<td>of which travel cost</td>
<td>77.6%</td>
<td>0.8</td>
</tr>
<tr>
<td>of which emergency cost</td>
<td>3.1%</td>
<td>-26.4</td>
</tr>
<tr>
<td>of which holding cost</td>
<td>19.3%</td>
<td>19.8</td>
</tr>
</tbody>
</table>

KMSAV and SAV. The geographical focus of the KM heuristic, and its relative inability to match demand distributions, leads to significantly less clusters than the other heuristics. For the KM heuristic, the most efficient way to reduce transportation and holding costs therefore is to create large clusters, which are frequently replenished. Finally, all heuristics have significantly lower emergency shipment costs than MIP-CG while meeting the capacity constraint, which indicates that truck capacity is not highly utilized compared to the solution from MIP-CG.

We exemplify the on-average characteristics by comparing the optimal solution of a single instance \(N = 15, h = 1.0, Q = 70\) with the KM and SAV solution. In Figure 2, the solutions are portrayed, with next to each cluster its associated shipment interval, and the objective value in the top right corner. Two observations stand out. First, the optimal solution is a combination of retailer clusters of both constructive heuristics. Second, zooming in on the three retailers in the upper right part of the graph, we observe that the SAV heuristic is not able to merge the three retailers into a single cluster. This behavior is caused by the algorithm’s myopic characteristic of only evaluating the consolidation of two and not more retailer clusters.

Summarizing, explicitly considering demand as in KMSAV and SAV when clustering retailers, leads to small clusters with higher shipment intervals, whereas the geographical focus of KM leads to large clusters with frequent deliveries.

Figure 2  Solution Structures by MIP-BPC (Left), KM (Middle), and SAV (Right)
7.4.2. Impact of Different Cost Components. In Figure 3, we investigate the effect of ignoring holding and emergency costs on the structure of the optimal solution portrayed in Figure 2. On the left of Figure 3, we see the same solution as in the left panel of Figure 2, whereas the middle and right solutions correspond to setting $h = 0$ and $e = 0$, respectively. Zooming in on the solution without holding costs (middle panel), we observe that direct deliveries become favorable for most retailers. This is because the reduction in fixed shipment costs achieved by replenishing retailers as infrequently as possible is higher than the effect of less variable shipment costs resulting from merging retailers into larger clusters. Moreover, when emergency shipment costs are ignored (right panel), we observe that retailer groupings increase. Note that the chance constraint on truck capacity still limits the size of the retailer groupings, and excess items not fitting on the truck are still delivered to the retailers by an external service provider but at zero costs. If excess items can be transported at zero costs, then less buffer space needs to be considered on the vehicle, which increases cluster sizes.

![Figure 3 Solution Structures of the Optimization Problem (left), in the Case Where Holding Costs Are Ignored (Middle), and in the Case Where Emergency Costs Are Ignored (Right)](image)

8. Conclusions and Future Research

We have presented a new approach to the stochastic inventory routing problem taking into account fixed replenishment intervals to groups of retailers mandated by, e.g., the retail industry to allow coordination with up- and downstream planning operations. Both our optimal solution and our heuristic solutions balance fixed as well as variable transportation costs, emergency shipment costs, and holding costs by allocating retailers to clusters that are replenished in fixed intervals.

We transformed the naturally integer chance-constrained model to a linear integer model, which we solve using column generation. We have provided an exact branch-price-and-cut method and several upper bounding procedures that are also used as an initial set of columns for the exact method. The efficiency of the exact method relies on a labeling algorithm tailored toward our setting and utilizes an approximate stochastic knapsack solution as a pruning mechanism. Whereas the exact method is able to solve, depending on the chosen parameter values, instances with up to 60 retailers, the hybrid heuristic achieves a performance that
is within 1% from optimality. This shows that for large-scale practical instances, the combination of our exact method (which can provide a lower bound) with the use of our heuristic methods show an excellent performance that is provably near-optimal.

An analysis of the cluster sizes and shipment intervals for the constructive heuristics revealed that all heuristics generate more clusters than is optimal, and that explicitly considering stochastic demand in the clustering process leads to small clusters with higher shipment intervals whereas a clustering based on geographic locations leads to large clusters with frequent deliveries mostly every period. Furthermore, we have shown how different cost components impact the structure of the optimal solution, which reveals the importance of considering all these cost components—transportation costs, emergency shipment costs, and holding costs—in the context of stochastic inventory routing problems.

The opportunities for further research are numerous. Many of these opportunities would not require structural adaptations of the presented model: for instance, one could consider delivery lead times that depend on the customer order in a tour and the corresponding transportation times, which requires modeling on a continuous time horizon. We believe that such extensions are valuable on its own, and our presented model can be used as a starting point.

More structural changes are required when, for instance, there is limited supply at the central warehouse, correlated demand between retailers and/or items are considered, vehicle tours are scheduled, or different shipment intervals in each retailer group are allowed. Limited supply at the central warehouse requires an allocation of available inventory to retailers as well as consideration of waiting times due to stock-outs at the central warehouse. Correlated demand increases the complexity of utilizing vehicle capacity and therefore also increases the complexity of clustering retailers into groups. In this paper, we have not scheduled vehicles and thus have assumed that each tour is performed by a different vehicle. By scheduling the tours, one can determine the optimal number of required vehicles, which is important if the shipments are performed by the company itself rather than by an external service provider. One can allow for different shipment intervals in each retailer group by, for example, considering so-called power-of-two policies described in the joint replenishment literature (see, e.g., Federgruen and Zheng 1992, Jackson et al. 1985), where each retailer is replenished in constant intervals that are power-of-two multiples of some base shipment interval. Thus, the number of parameters that needs to be optimized increases significantly.
Appendix A: Notation

<table>
<thead>
<tr>
<th>Table 6</th>
<th>Overview of main notation used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sets:</td>
<td></td>
</tr>
<tr>
<td>$\mathcal{N}$</td>
<td>set of retailers</td>
</tr>
<tr>
<td>$\mathcal{N}^0$</td>
<td>set including all retailers and the central warehouse</td>
</tr>
<tr>
<td>$\mathcal{K}$</td>
<td>set of retailer groups</td>
</tr>
<tr>
<td>$\mathcal{A}$</td>
<td>complete edge set</td>
</tr>
<tr>
<td>Indices:</td>
<td></td>
</tr>
<tr>
<td>$N$</td>
<td>number of retailers</td>
</tr>
<tr>
<td>$i,j$</td>
<td>index for retailers</td>
</tr>
<tr>
<td>$K$</td>
<td>number of retailer groups</td>
</tr>
<tr>
<td>$k$</td>
<td>index for retailer groups</td>
</tr>
<tr>
<td>Parameters:</td>
<td></td>
</tr>
<tr>
<td>$D_i$</td>
<td>random variable for the demand per period at retailer $i$</td>
</tr>
<tr>
<td>$\mathbb{E}[D_i]$</td>
<td>mean demand per period at retailer $i$</td>
</tr>
<tr>
<td>$\text{VAR}[D_i]$</td>
<td>variance of the demand per period at retailer $i$</td>
</tr>
<tr>
<td>$D_i(T_k)$</td>
<td>random variable for the demand for retailer $i$ during $T_k$ periods</td>
</tr>
<tr>
<td>$\alpha^*$</td>
<td>target service level at retailer $i$</td>
</tr>
<tr>
<td>$\gamma^*$</td>
<td>target probability that the vehicle’s capacity for a replenishment is not exceeded</td>
</tr>
<tr>
<td>$Q$</td>
<td>capacity of all vehicles</td>
</tr>
<tr>
<td>$IL^+_i(t,S_i)$</td>
<td>stock on hand at the end of period $t$ at retailer $i$ depending on the base-stock level $S_i$</td>
</tr>
<tr>
<td>$G$</td>
<td>graph</td>
</tr>
<tr>
<td>Decision variables:</td>
<td></td>
</tr>
<tr>
<td>$S$</td>
<td>vector of base-stock levels</td>
</tr>
<tr>
<td>$S_i$</td>
<td>base-stock level for retailer $i$</td>
</tr>
<tr>
<td>$T$</td>
<td>vector of shipment intervals to all retailer groups</td>
</tr>
<tr>
<td>$T_k$</td>
<td>shipment interval for group $k$</td>
</tr>
<tr>
<td>$R_i$</td>
<td>review period for retailer $i$ belonging to group $k$ ($R_i = T_k$)</td>
</tr>
<tr>
<td>$Y$</td>
<td>$(N \times N \times K)$ matrix containing binary decision variables $y_{ijk}$</td>
</tr>
<tr>
<td>$y_{ijk}$</td>
<td>binary decision variable that equals 1 if retailer $j$ is visited after retailer $i$ in group $k$, and 0 otherwise</td>
</tr>
<tr>
<td>$X$</td>
<td>$(N \times K)$ matrix containing binary decision variables $x_{ik}$</td>
</tr>
<tr>
<td>$x_{ik}$</td>
<td>binary decision variable that equals 1 if retailer group $k$ contains retailer $i$, and equal 0 otherwise</td>
</tr>
<tr>
<td>Cost components:</td>
<td></td>
</tr>
<tr>
<td>$TC$</td>
<td>total expected cost per period</td>
</tr>
<tr>
<td>$W$</td>
<td>fixed shipment costs per shipment</td>
</tr>
<tr>
<td>$c_{ij}$</td>
<td>travel cost between retailer $i$ and $j$</td>
</tr>
<tr>
<td>$w_k$</td>
<td>variable shipment cost per shipment to retailer group $k$</td>
</tr>
<tr>
<td>$e$</td>
<td>emergency shipment cost per emergency shipment to one retailer group</td>
</tr>
<tr>
<td>$h$</td>
<td>holding cost per unit and time unit</td>
</tr>
</tbody>
</table>

Appendix B: Proof of Lemma 4

Since $D_i$ and $D_j$ are gamma distributed independent random variables with the same scale parameter, the distribution of $D_i + D_j$ is a gamma distribution with shape parameter $\kappa_i + \kappa_j$ and scale parameter $\theta$. The distribution of $D_i + \mu_j$ is a
shifted gamma distribution with density

\[
f(x) = \begin{cases} 
0 & x \leq \mu_j, \\
\frac{1}{\Gamma(k_i)} \frac{1}{\theta^k_i} \frac{1}{x^{k_i + 1}} e^{-\frac{x - \mu_j}{\theta}} & x > \mu_j.
\end{cases}
\]  

(30)

There exists an \( x_0 \) such that for all \( x \geq x_0 \) the following inequality holds,

\[
\left( \frac{1}{\theta} \right)^{k_i} \frac{1}{\Gamma(k_i + k_j)} x^{k_i + k_j - 1} e^{-\frac{x - \mu_j}{\theta}} \geq \frac{1}{\Gamma(k_i)} (x - \mu_j)^{k_i - 1} e^{-\frac{(x - \mu_j)}{\theta}},
\]

(31)

because the grade of the polynomial on the left hand side is larger than on the right hand side. This yields

\[
f_{D_i + D_j}(x) \geq f_{D_i + \mu_j}(x).
\]

(32)

Due to (32) we get for all \( x \geq x_0 \)

\[
\int_x^{+\infty} f_{D_i + D_j}(s) \, ds \geq \int_x^{+\infty} f_{D_i + \mu_j}(s) \, ds,
\]

(33)

which is equivalent to

\[
1 - P(D_i + D_j \leq x) \geq 1 - P(D_i + \mu_j \leq x).
\]

(34)

Thus,

\[
P(D_i + D_j \leq x) \leq P(D_i + \mu_j \leq x) \quad \forall x \geq x_0.
\]

(35)

Let us denote with \( x_a \) (\( y_a \)) the \( a \% \) quantile of the cumulative distribution function of \( D_i \) \((D_i + D_j)\). Then we have

\[
P(D_i \leq x_a) = a = P(D_i + D_j \leq y_a).
\]

(36)

It follows that

\[
P(D_i \leq x_a) = P(D_i + \mu_j \leq x_a + \mu_j) = P(D_i + D_j \leq y_a).
\]

(37)

Since for \( x \geq x_0 \),

\[
P(D_i + D_j \leq x) \leq P(D_i + \mu_j \leq x),
\]

(38)

there exists an \( \tilde{a}(\kappa_i, \kappa_j, \theta) \) (dependent on \( x_0 \)) such that for all \( a > \tilde{a}(\kappa_i, \kappa_j, \theta) \)

\[
P(D_i + D_j \leq x_a + \mu_j) \leq P(D_i + \mu_j \leq x_a + \mu_j) = P(D_i + D_j \leq y_a)
\]

(39)

Since the CDF of the gamma distribution is monotonously increasing we get \( y_a \geq x_a + \mu_j \) what was to be shown.
References


