European Journal of Operational Research 332 (2026) 764–773

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Discrete Optimization

Branch-and-repair for the stochastic three-dimensional bin selection problem: A multi-stage stochastic programming application

Pirmin Fontaine [a,b]

[a] Catholic University of Eichstätt-Ingolstadt, Ingolstadt School of Management, Auf der Schanz 49, 85049, Ingolstadt, Germany
[b] Catholic University of Eichstätt-Ingolstadt, Mathematical Institute for Machine Learning and Data Science, Ingolstadt, Germany

ARTICLE INFO

Keywords:
Packing
Branch-and-repair
Multi-stage stochastic programming
E-commerce
Three-dimensional bin selection

ABSTRACT

Increasing e-commerce has been one of the major trends in the last decades. One of the key elements is the packing of items where the company has to decide which parcel type to choose for packing all items of an order. To reduce the unused space in parcels, the essential part is the available parcel type portfolio. Since order demand is uncertain, it is important to account for this uncertainty in the strategic decision process. Therefore, we address the strategic design of the parcel type portfolio under stochastic demand and introduce the stochastic three-dimensional bin selection problem (S3D-BSP).

We formulate the S3D-BSP as a multi-stage stochastic program in which recourse decisions allow for reordering parcels of the chosen portfolio and balance inventory and procurement decisions. To solve larger instances, we introduce a branch-and-repair method. Specifically, we develop a fast approximated and a slow but optimal repair strategy and discuss how to balance the trade-off between those two. In the numerical study, we show the efficiency of our approach and show that optimal portfolios can largely vary depending on the demand structure in a case study based on real-world data.

1. Introduction

E-commerce is continuously growing, resulting in an increasing number of parcels every year that is shipped around the globe. Pitney Bowes (2023) report a global parcel volume of 161 billion parcels in 2022. This is an increase of 150% from 2016. This increase is further intensified by the ongoing urbanization. Therefore, research on urban logistics is continuously growing and tries to improve the efficiency in both parcel and freight transportation (Crainic et al., 2009). One major challenge is the poor utilization of resources resulting from not well-packed trucks, containers, and wrong-sized parcels (Fontaine & Minner, 2023). According to Esser and Kurte (2020) unused space in parcels can reach up to 50% and Insights (2018) even estimate more than 60% of unused space on average in certain product categories which has to be filled with filling material that causes additional waste. The European Union is currently in the process of updating its regulations on packaging and packaging waste. One article in this proposal states that empty space is not allowed to exceed 40% anymore (European Union, 2024). Thus, as soon as these regulations are in place, companies will be forced to improve their packaging strategies. To assist companies in packing, parcel delivery providers have started developing tools to help pack their items. For example, DHL's AI tool OptiCarton recommends packages and packing patterns (DHL, 2022) and UPS offers a cube optimization service (UPS, 2021).

Additionally, concepts for last-mile delivery, such as two-tier city logistics systems (Crainic et al., 2009), will require more handling effort in the future since parcels are transferred at intermediate hubs between the two tiers. Decreasing the unused space in parcels or containers would allow to pack goods more efficiently and reduce the number of needed vehicles.

To efficiently assign orders to parcels or shipments to containers, the available portfolio of parcels and containers is important. Therefore, e-commerce retailers must carefully select the parcel types they want to keep in stock. In the case of city logistics, this transfers to the question of finding standardized so-called π-containers (Crainic & Montreuil, 2016). While such a standardization improves also the efficiency of city logistics systems, the sizes and the variety of such π-containers are still unclear (Montreuil et al., 2015).

In the deterministic case, the question of the number and size of the used parcel types in e-commerce has been recently answered (Fontaine & Minner, 2023). However, according to the Beumer Group, a manufacturer of intralogistics systems, the dynamics of e-commerce pose one of the major challenges in planning systems for the appropriate parcel mix (Johansen, 2024). This implies that tools are necessary that

E-mail address: pirmin.fontaine@ku.de

https://doi.org/10.1016/j.ejor.2026.01.035
Received 28 March 2025; Accepted 22 January 2026
Available online 24 January 2026
0377-2217/© 2026 The Author. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

take the stochasticity in demand over longer planning horizons into account.

Since current literature ignores this uncertainty, we aim to improve efficiency and sustainability by analyzing the trade-off between inventory holding, costs of unused space, and the costs of variety in a multi-period setting under stochastic demand. For e-commerce retailers, a smaller portfolio implies that fewer parcel types need to be kept in stock and handled by the supply chain department. On the other side, a larger portfolio increases the chance of reducing unused space in parcels. Additionally, the dynamic nature of the problem resulting in replenishment policies allows the assignment of orders to a larger parcel type if the best-fitting parcel type is not in stock even though it is part of the parcel portfolio.

From a modeling perspective (Section 3), we address this gap by introducing the S3D-BSP and formulating it as a multi-stage stochastic program. In this problem setting, we select a portfolio of three-dimensional parcel types out of a larger set of available parcel types. Then, over the planning horizon, these parcel types can be dynamically procured to pack orders of three-dimensional items into the parcels on stock.

Methodologically (Section 4), we develop an exact solution method building on branch-and-repair (Fontaine & Minner, 2023) and branch-and-check (Thorsteinsson, 2001). In this approach, the problem is decomposed into a master and a subproblem. In the deterministic setting introduced by Fontaine and Minner (2023), this decomposition leads to an uncapacitated transportation problem as master problem. Although the S3D-BSP results in a more complex, multi-period, and inventory-constrained (i.e., capacitated) problem compared to the deterministic three-dimensional bin selection problem (3D-BSP), we prove that many binary variables can be relaxed. Further, we show that increasing the number of stages and scenarios does not impact the number of subproblems that need to be evaluated. This implies that branch-and-repair can further help to solve multi-stage stochastic programs efficiently, where the decision problem in each scenario and stage is hard to solve. We introduce two different repair strategies which differ in quality and run time. We show the benefits of not always aiming for high-quality solutions in the repair process while retaining optimality in the overall procedure.

In the numerical study (Section 5), we generate new data sets for the S3D-BSP and show the scalability of the method by increasing the size of the scenario tree and the number of underlying three-dimensional bin packing problem (3D-BPP).

From a managerial perspective, we use real-world data sets from the MSOM data-driven challenge (Guo et al., 2024) and show in Section 6 that portfolios significantly differ depending on the demand and the importance of multi-stage stochastic programming solutions.

In summary, the main contributions are (1) the introduction of the S3D-BSP, (2) the formulation as a multi-stage stochastic program, (3) the development of a branch-and-repair method for multi-stage stochastic programs also under capacitated problem settings, (4) the demonstration of its performance and a numerical study, and (5) managerial implications based on real data.

The remainder of the paper is structured as follows. First, we review the related literature. Then, in Section 3, we define the problem setting and describe the multi-stage stochastic program. Branch-and-repair is introduced in Section 4. Then, the numerical study and the case follow before ending with a conclusion.

2. Literature review

In this section, we first review the bin packing literature. Then, we discuss publications on multi-stage stochastic programs with particular focus on exact solution methods. We end this review with literature on exact decomposition techniques.

2.1. Bin packing

The literature on bin packing problems has a long history and many variants and applications have been investigated, also in combination with other problems (e.g., Zhang et al., 2022, for a vehicle routing problem). For a detailed overview and a classification of bin packing problems, we refer to Wäscher et al. (2007). One characteristic that defines these packing problems is the dimension of the items and boxes which is three-dimensional in our setting. Therefore, we focus on this class and refer to Delorme et al. (2016), Castro and Oliveira (2011), and Iori et al. (2021) for one- and two-dimensional settings. For the three-dimensional packing problem, Ali et al. (2022) recently summarized both on-line and off-line methods and detail constraints that are considered in the 3D-BPP literature. Further, Nascimento et al. (2021) and Gzara et al. (2020) introduce several formulations for practical constraints. In our setting, we use a stylized version of the problem but want to highlight that both problem setting and solution methodology can be easily adapted to different variants and integrate more constraints.

A second important factor in the S3D-BSP is the heterogeneity of items and boxes. When packing heterogeneous items into a selected subset of heterogeneous boxes with the objective to minimize unused space, this is categorized as a bin packing-type problem (Wäscher et al., 2007). In this context, Chen et al. (1995) study the three-dimensional container loading problem (CLP), which is classified as a Residual Bin Packing Problem in the taxonomy of Wäscher et al. (2007). A second important feature in the CLP is that rotation of items is allowed, which is not the case for many publications of the classical 3D-BPP (e.g., Martello et al., 2000). However, rotating items when being packed is very important for e-commerce applications and therefore included into our model. For the CLP, Paquay et al. (2016) introduce an efficient mixed-integer linear program (MILP), which is further enhanced through valid inequalities and a reformulation in Fontaine and Minner (2023). Due to its complexity, also many heuristics have been introduced. For example, šafak and Erdoğan (2023) recently developed an adaptive large neighborhood search. Additionally, Tsao et al. (2024) consider the multiple bin-size bin packing problem in the case of incompatible product categories. All these works have in common with our work that the unused space is minimized in a three-dimensional case. However, we differ since we assume that only a limited number of different bin types should be selected. Further, the variable-sized bin packing problem (e.g., Friesen & Langston, 1986) is related yet different to our problem setting. While again different bins are available, there is no incentive to reduce the subset of bins that should be considered. Salma and Ahmed (2011) consider a three-dimensional variant where the objective is to find a minimum number of bins to place items (mattresses) into.

However, the question of how to design a portfolio of bin types has rarely been addressed so far. Dowsland et al. (2007) consider a two-dimensional setting. They use a simulated annealing algorithm to determine the portfolio and solve a pallet loading problem as subproblem. While Brinker and Gündüz (2016) consider the three-dimensional case, they only solve the portfolio design and the packing problem separately. For a single-item case, Vieira et al. (2021) formulate a p-median problem that is used packing shoes into boxes. Instead, we assume a much more challenging three-dimensional packing problem and introduce an exact algorithm for the integrated problem setting. The only integrated problem that considers the packing of multiple three-dimensional items into bins and designs the portfolio is recently introduced by Fontaine and Minner (2023). For solving the problem, the authors introduce branch-and-repair and show the effectiveness of the decomposition method even for large instance. Still, they only consider deterministic demand. This paper not only assumes stochastic demand but also considers a multi-period planning problem where the bins of the portfolio can be replenished in each period. This further results in a trade-off between procurement and inventory holding costs and is modeled as a multi-stage stochastic program compared to an MILP in Fontaine and Minner (2023).

2.2. Multi-stage stochastic programming

Multi-stage stochastic programs are well-studied and known to be very challenging to solve (see, e.g., Birge & Louveaux, 2011; Kall & Wallace, 1994). Because of this complexity, not only a wide range of heuristics but also exact solution methods have been proposed. Birge (1985) develops a decomposition method for multi-stage stochastic linear programs. Daryalal et al. (2022) propose dual decision rules for multi-stage problems with integer recourse decisions. Recently, Kayacık et al. (2025) introduced an L-Shaped method to derive the optimal number of decisions in a multi-stage setting and show how to reduce the number of non-anticipativity constraints. The L-shaped method (Van Slyke & Wets, 1969) is known for decomposing linear programs and can be extended to integer programming problems (Laporte & Louveaux, 1993; Laporte et al., 2002).

However, in these methods, the focus often lies on the complexity of the master problem, e.g., tightening the formulation (Kayacık et al., 2025). Another approach introduced by Keutchayan et al. (2020) shows the effect of reducing the generated scenario tree in stochastic programs. Contrary to this, this paper develops a methodology that is capable of dealing with a large number of complex problems in the recourse decision. Explicitly, we consider in each stage and each scenario a large number of 3D-BPPs, which is a challenging NP-hard problem itself.

2.3. Decomposition methods

Besides its application in multi-stage stochastic programs, decomposition methods have been applied to various applications and two-stage settings (see, e.g., Çelik et al., 2025). We build on the idea of branch-and-repair that recently has been introduced by Fontaine and Minner (2023). Branch-and-repair uses the ideas of branch-and-check methods (Thorsteinsson, 2001) and logic-based Benders decomposition (e.g., Hooker, 2007; Hooker & Ottosson, 2003) where the full model is decomposed into a relaxed master problem and a subproblem. The relaxed master problem ignores a subset of constraints, which then defines the subproblem, and then checks the feasibility of the subproblem. In case this is true, a feasible solution is found. Otherwise a cut is added. Compared to the classical Benders decomposition (Benders, 1962) and the L-Shaped method (Van Slyke & Wets, 1969), branch-and-repair does not only check for feasibility of the subproblem. Instead, it tries to repair the provided solution of the master problem to efficiently generate upper bounds. However, Fontaine and Minner (2023) consider a deterministic single-period problem setting that results in an uncapacitated master problem. Contrary, the multi-period stochastic version in this paper results in a capacitated master problem. This requires the development of a new repair mechanism for a much more challenging problem setting.

Particularly for cutting and packing problems, decomposition methods have already shown their potential to be efficient exact methods. Côté et al. (2014) solve the strip packing problem using a Benders decomposition method. For the two-dimensional bin packing problem, Côté et al. (2021) present a combinatorial Benders decomposition. The main contribution in this work is the usage of minimal infeasible subsets of items to lift the generated cuts.

Finally, a large variety of acceleration techniques for Benders decomposition exist. For a recent overview, we refer to Rahmaniani et al. (2017).

3. Problem setting

We consider a packing problem where a company needs to pack orders of different items over a given planning horizon. Such a planning horizon depends on the company and industry but might be, e.g., several months or a season. For packing and shipping those orders, a parcel, from now on called bin type, is selected and all items of one order are packed into the selected bin. For doing so, the company is interested in a long-term commitment regarding a bin type portfolio (a subset of bin types) which is used over the whole planning horizon. This long-term commitment reflects the necessity to set up contracts with the parcel supplier and standardizes the packing and shipping processes. Besides the question of the bin type portfolio, the dynamic nature of the problem leads to the task of procuring bins out of the bin type portfolio to fill up current stock. Thus, bin types can be procured, e.g., once per month, to refill the stock. This results not only in a trade-off between procurement and inventory holding costs, known from inventory control (e.g., Axsäter, 2015) but also between diversification for better fitting bin types (resulting in less unused space) and pooling effects (Alptekinoğlu et al., 2013) because of the stochastic demand.

The orders that need to be packed consist of a set of three-dimensional rectangular items and the stochastic demand, the quantity of these type of orders. This means that we assume if five customers order the same-sized items, this is considered as one order of quantity five. In each period, the goal is to pack the orders efficiently, i.e., reducing the unused space in one selected three-dimensional rectangular bin. However, the stochastic demand and the lot sizing decisions can lead to inefficient assignments of orders to bins if the best-fitting bin is not on stock. In case the perfect-fitting bin type is out of stock, a larger bin type needs to be selected which results in more unused space in the parcel, implying more unsustainable transportation.

Overall, the decision maker wants to find the perfect portfolio to minimize the total costs of delivery cost, procurement cost, inventory holding cost and cost of variety of the bin type portfolio.

Please note that we assume all items of an order are packed into a single bin, which aligns with common practices in e-commerce logistics. Although customers may experience their orders being split into multiple deliveries, this is typically due to external factors, such as items being stored in different warehouses or some products being temporarily unavailable and shipped later (e.g., Brylla & Walsh, 2022; Zhang et al., 2019). These splitting decisions occur prior to the packing process. Once the items for a single delivery are consolidated, the packing into a single bin is executed. Furthermore, Fontaine and Minner (2023) show that order splitting has minimal impact on the unused space in parcels.

3.1. Stochastic three-dimensional bin selection problem

To address the problem setting presented in the previous section, we introduce the S3D-BSP as a multi-stage stochastic program. Since our modeling approach and solution methodology can use different variants of the packing problem, we first introduce the general planning problem, then we detail the three-dimensional packing problem that we assume.

We consider a planning horizon defined by a set of periods T, which is equivalent to the stages in the multi-stage stochastic program and the periods where procurement decisions can be placed. Then, J defines the set of rectangular bin types (parcels) and O the set of orders. Each of these orders $o \in O$ consists of a set of rectangular items I_o that have to be packed into a selected bin. For packing those items, we assume the standard assumptions of three-dimensional packing with rotation: (1) items are allowed to be rotated along the three axes, (2) the bin has to enclose all packed items, and (3) items are not allowed to overlap.

Both the bin types $j \in J$ and the items $i \in I_o$ of the orders $o \in O$ are specified by a length L_j, a width W_j, and a height H_j and l_{io}, w_{io}, and h_{io}, respectively. To include the stochastic nature of the demand over the planning horizon, we assume a set of scenarios S with probability π_s resulting in a demand d_{ots} for each order $o \in O$, stage $t \in T$, and scenario $s \in S$. For each scenario s, $\mathbf{P}_t(s)$ defines the scenario history until stage t. This means that if $\mathbf{P}_t(s) = \mathbf{P}_t(s')$ for two scenarios $s \neq s' \in S$, these scenarios are the same until stage t (Bertazzi & Maggioni, 2018).

The main decision is to determine the optimal set of bin types that is used over the full planning horizon. Therefore, the binary decision $\hat{n}_j \in \{0,1\}$ indicates if a bin type is part of the portfolio or not. Decision variable q_{jts} defines the quantity of bin type j that is procured in stage t of scenario s, and the binary decision $\hat{q}_{jts} \in \{0,1\}$ if a procurement decision is placed or not. Further, p_{jts} tracks the available inventory of each

bin type j. Additionally, the integer decision variable n_{ojts} defines the number of orders o that is assigned to bin type j in stage t in scenario s. While q_{jts} and p_{jts} also represent integer quantities, the problem setting allows continuous decision variables. This is not the case for n_{ojts}. Finally, let $3DBPP(o,t,s)$ define the constraints and decision variables of a packing problem that is used in the problem setting (see Section 3.2 for the variant that is used in this paper). These constraints have to be satisfied for each order in each stage and scenario.

The objective function includes costs of unused space c^S, costs of variety c^V, inventory holding costs c^I, and procurement costs c^O. Please note that in the first stage only the portfolio $\hat{n}_j$ is determined and the initial inventory is set. As a result, the first-stage objective includes only the cost-of-variety term. Therefore, we directly present the complete deterministic equivalent model and define the S3D-BSP as follows:

$$\min \sum_{s\in S}\sum_{t\in T}\pi_s\left(c^S\sum_{o\in O}\left(\sum_{j\in J}L_jW_jH_jn_{ojts}-\sum_{i\in I_o}l_{io}w_{io}h_{io}d_{ots}\right)+c^I\sum_{j\in J}p_{jts}+c^O\sum_{j\in J}\hat{q}_{jts}\right)+c^V|O|\sum_{j\in J}\hat{n}_j \tag{1}$$

$$\text{s.t.}\quad 3DBPP(o,t,s) \qquad \forall o\in O, s\in S, t\in T \tag{2}$$

$$\sum_{j\in J}n_{ojts}=d_{ots} \qquad \forall o\in O, s\in S, t\in T \tag{3}$$

$$\hat{q}_{jts}\le\hat{n}_j \qquad \forall j\in J, s\in S, t\in T \tag{4}$$

$$q_{jts}\le M\hat{q}_{jts} \qquad \forall j\in J, s\in S, t\in T \tag{5}$$

$$p_{j,t-1,s}+q_{jts}-\sum_{o\in O}n_{ojts}=p_{jts} \qquad \forall j\in J, s\in S, t\in T \tag{6}$$

$$n_{ojts'}=n_{ojts''} \qquad \forall o\in O, j\in J, t\in T, s', s''\in S|\mathbf{P}_t(s')=\mathbf{P}_t(s'') \tag{7}$$

$$p_{jts'}=p_{jts''} \qquad \forall j\in J, t\in T, s', s''\in S|\mathbf{P}_t(s')=\mathbf{P}_t(s'') \tag{8}$$

$$q_{jts'}=q_{jts''} \qquad \forall j\in J, t\in T, s', s''\in S|\mathbf{P}_t(s')=\mathbf{P}_t(s'') \tag{9}$$

$$\hat{q}_{ts'}=\hat{q}_{ts''} \qquad \forall t\in T, s', s''\in S|\mathbf{P}_t(s')=\mathbf{P}_t(s'') \tag{10}$$

$$q_{jts}, p_{jts}\ge 0 \qquad \forall j\in J, s\in S, t\in T \tag{11}$$

$$\hat{n}_j, \hat{q}_{jts}\in\{0,1\} \qquad \forall j\in J, s\in S, t\in T \tag{12}$$

$$n_{ojts}\in\mathbb{Z}^{\ge 0} \qquad \forall o\in O, j\in J, s\in S, t\in T \tag{13}$$

We minimize the trade-off between the costs for unused space, inventory holding costs, procurement costs, and the costs of variety. Constraints (2) ensure that the packing constraints are satisfied. Constraints (3) assign orders to bin types and ensure all orders are packed. A bin procurement decision can only be placed if the bin is part of the portfolio (Constraints (4)), and only if a procurement decision is placed, bins can be replenished (Constraints (5)). Inventory balance is ensured through constraints (6). Through the non-negativity of the decision variables, these constraints further ensure that the capacity constraints of the bin types on stock are not violated. Finally, classical non-anticipativity constraints (see, e.g., Bertazzi & Maggioni, 2018) are included via constraints (7)-(10) and constraints (11)-(13) define the domains of the decision variables.

3.2. Three-dimensional bin packing problem

For the 3D-BPP (see Fontaine & Minner, 2023), we further define $L_0=\max_{j\in J}\{L_j\}, W_0=\max_{j\in J}\{W_j\}, H_0=\max_{j\in J}\{H_j\}$ as the maximum of each of the dimensions. The reason behind this is that the authors show that normalizing the length, width, and height of the bins with their maximum dimensions, allows to use position variable values between zero and one. This avoids big M values and reduces run time. For better readability, we omit in the following the index o in set I and the indices o, t, and s which would be necessary for all decision variables. First, the continuous decision variables $(x_i, y_i, z_i)\in[0,1]^3$ define the relative position of the left-front-bottom corner of item i in its bin. Similarly, $(x_i^r, y_i^r, z_i^r)\in[0,1]^3$ define the relative position of the right-back-upper corner. The binary decision $r_{ij}\in\{0,1\}$ defines if item i is placed into bin j and s_j if bin j is used for the order in the stage and scenario. To forbid overlapping of two items $i,k\in I$ that are placed into the same bin, the binary decisions a_{ik}^x (a_{ik}^y,a_{ik}^z)$\in\{0,1\}$ indicate if item i is on the right of (behind, above) item k in the x-(y-,z-)dimension. For rotating items, let $C=\{1,2,3\}$ define the 3 dimensions (x,y,z) of the bin and $D=\{1,2,3\}$ the 3 dimensions (x,y,z) of the item. Then, the $t_{i,c,d}\in\{0,1\}$ state the orientation of item i in its selected bin. For example, $t_{i,3,1}=1$ indicates that the x-dimension (1) of item i with length l_{io} is placed along the z-dimension (3) of the selected bin.

Taking the number of assigned orders to a bin n_{ojts} from the full model, the 3D-BPP can be formulated as

$$\min \sum_{j\in J}L_jW_jH_jn_{ojts}-\sum_{i\in I}l_iw_ih_id_{ots} \tag{14}$$

$$\text{s.t.}\quad x_i^r-x_i=t_{i,1,1}\frac{l_i}{L_0}+t_{i,1,2}\frac{w_i}{L_0}+t_{i,1,3}\frac{h_i}{L_0} \qquad \forall i\in I \tag{15}$$

$$y_i^r-y_i=t_{i,2,1}\frac{l_i}{W_0}+t_{i,2,2}\frac{w_i}{W_0}+t_{i,2,3}\frac{h_i}{W_0} \qquad \forall i\in I \tag{16}$$

$$z_i^r-z_i=t_{i,3,1}\frac{l_i}{H_0}+t_{i,3,2}\frac{w_i}{H_0}+t_{i,3,3}\frac{h_i}{H_0} \qquad \forall i\in I \tag{17}$$

$$x_i^r\le\sum_{j\in J}\frac{L_j}{L_0}r_{ij} \qquad \forall i\in I \tag{18}$$

$$y_i^r\le\sum_{j\in J}\frac{W_j}{W_0}r_{ij} \qquad \forall i\in I \tag{19}$$

$$z_i^r\le\sum_{j\in J}\frac{H_j}{H_0}r_{ij} \qquad \forall i\in I \tag{20}$$

$$a_{ik}^x+a_{ki}^x+a_{i,k}^y+a_{ki}^y+a_{ik}^z+a_{ki}^z\ge r_{ij}+r_{kj}-1 \qquad \forall i,k\in I, j\in J; i<k \tag{21}$$

$$\sum_{j\in J}r_{ij}=1 \qquad \forall i\in I \tag{22}$$

$$r_{ij}\le n_{ojts} \qquad \forall i\in I, j\in J \tag{23}$$

$$r_{ij}\le s_j \qquad \forall i\in I, j\in J \tag{24}$$

$$\sum_{j\in J}s_j\le 1. \tag{25}$$

$$\sum_{c\in C}t_{i,c,d}=1 \qquad \forall i\in I, d\in D \tag{26}$$

$$\sum_{d\in D}t_{i,c,d}=1 \qquad \forall i\in I, c\in C \tag{27}$$

$$x_k^r\le x_i+(1-a_{ik}^x) \qquad \forall i,k\in I \tag{28}$$

$$x_i+\frac{1}{L_0}\le x_k^r+a_{ik}^x \qquad \forall i,k\in I \tag{29}$$

$$y_k^r\le y_i+(1-a_{ik}^y) \qquad \forall i,k\in I \tag{30}$$

$$y_i+\frac{1}{W_0}\le y_k^r+a_{ik}^x \qquad \forall i,k\in I \tag{31}$$

$$z_k^r\le z_i+(1-a_{ik}^z) \qquad \forall i,k\in I \tag{32}$$

$$r_{ij}, t_{i,c,d}, a_{ik}^x, a_{ik}^y, a_{ik}^z\in\{0,1\} \qquad \forall i,k\in I, j\in J, c\in C, d\in D \tag{33}$$

$$x_i, y_i, z_i, x_i^r, y_i^r, z_i^r\ge 0 \qquad \forall i\in I \tag{34}$$

The objective function (14) subtracts the constant volume of all times from the volume of the selected bin and, therefore, minimizes the unused space in the selected bin. Constraints (15), (16), and (17) link the position of left and right corner for each item depending on the chosen orientation of the item for the x-, y-, and z-dimension. Due to the normalization of the position variables the dimensions of the items placed into the bin are normalized accordingly. To ensure feasibility, Eqs. (18)–(20) restrict the location of the right corner by the normalized size of the chosen bin. Constraints (21) forbid overlapping of two items. Eq. (22) guarantee that each item is packed and (23) restricts this choice to bins that are used in the master problem. Due to the e-commerce setting, we ensure that all items are packed into a single bin by constraints (24) and

(25). Packing items of one order into separate bins results in e-commerce mainly from products not being available at the same time or at the same distribution center (Biggs, 2021). Constraints (26) and (27) define the rotation of the items by ensuring that each side of an item is chosen exactly once and placed into a single direction. Further, Eqs. (28)–(32) set the overlapping decision variables. In the x-dimension, constraints (28)–(29) indicate that the right corner of item k has to be smaller than the left corner of item i if item i is placed right of item k. The same idea is used for the y- and z-dimension in Eqs. (30)–(32). While in general each of the dimensions requires two constraints, in one dimension (we chose the z-dimension), the second constraint is redundant and can be removed (see, e.g., Fontaine & Minner, 2023; Paquay et al., 2016). The model concludes with the domains of the decision variables in (33) and (34).

Rotation of items is a standard assumption in e-commerce. Nonetheless, including a more restrictive packing problem with the so-called "this side up" orientation for item i can be ensured by setting $t_{i,1,3} = 0$, $t_{i,2,3} = 0$, $t_{i,3,1} = 0$, $t_{i,3,2} = 0$, and $t_{i,3,3} = 1$ (Alonso et al., 2016; Fontaine & Minner, 2023).

From a classification perspective, the structure of the introduced packing subproblem is as follows. It combines two components: (i) selecting exactly one bin type from a candidate bin type portfolio and (ii) packing all items of an order into the selected bin. Hence, conditional on a fixed bin type, the model reduces to a Single Container Loading Problem that classifies as Single Knapsack Problem in the taxonomy of Wäscher et al. (2007). The additional bin-type selection step embeds this Single Container Loading Problem into a bin-selection setting.

4. Branch-and-repair

To solve the S3D-BSP, we develop a general branch-and-repair method (Fontaine & Minner, 2023) for multi-stage stochastic programs that further uses problem-specific structures for run time improvements. The general idea is to relax the S3D-BSP by leaving out the 3D-BPP constraints. When solving this relaxed master problem (see Section 4.1) in a branch-and-cut tree, a subroutine checks at each integer node if the 3D-BPP constraints are satisfied or not. If this is true, a feasible solution is found. If not, combinatorial cuts that forbid this integer solution are added to the branch-and-cut tree. Moreover, a repair method tries to find a feasible solution based on the current infeasible solution and insert it into the branch-and-cut tree to generate upper bounds (Section 4.3). Details on used existing and new improvement strategies are then detailed in Section 4.4.

4.1. Relaxed master problem

We define the relaxed master problem of the S3D-BSP as objective function (1) subject to constraints (3)–(12). Thus, we relax the original problem by ignoring the 3D-BPP constraints (2). Moreover, the integrality constraint of the decision variables n_{ojts} can be relaxed.

Remark 1. The domain of the integer decision variables n_{ojts} for all orders o and all bin types j in stage t of scenario s in the relaxed master problem can be relaxed to

$$n_{ojts} \geq 0 \qquad \forall o \in O, j \in J, t \in T, s \in S. \tag{35}$$

Fontaine and Minner (2023) show that the deterministic version of the relaxed master problem is an uncapacitated transportation problem. In this paper, this results in a capacitated transportation problem for each scenario and stage. This still results in an integer solution in case of relaxation. Note that this relaxation is not possible in the full model of Section 3.1 but a result of the decomposition. Therefore, the choice of the decomposition is an important factor of the performance of branch-and-repair.

4.2. Integer node subroutine and improvements

In each integer node of the relaxed master problem, we check if the chosen assignment of orders to bin types is feasible or not. Thus, if the 3D-BPP constraints (2) are satisfied for all demands, scenarios, and stages or not. In the latter case, we try to repair this solution. The outline of this procedure is shown in Algorithm 1.

Algorithm 1: Integer node procedure.

```
   Input: solution n_ojts, n̂_j of current integer node
1  Γ = {}, Ψ_repair = false
2  for s ∈ S, t ∈ T, o ∈ O, j ∈ J do
3      if n_ojts > 0 then
4          if θ_oj = false then
5              φ_oj ← solve 3D-BPP of order o and bin j
6              θ_oj = true
7              if φ_oj = false then
8                  add constraint n_ojts = 0   ∀t ∈ T, s ∈ S
9          if φ_oj = false then
10             Ψ_repair = true
11             Γ = Γ ∪ {(o, j, t, s)}
12 end
13 if Ψ_repair = true then
14     if repairOrderAssignment(Γ) = true then
15         Insert repaired solution
```

The 3D-BPP needs to be evaluated for each stage in each scenario for all bin types that are assigned to an order ($n_{ojts} > 0$). To avoid unnecessary evaluations, we use a boolean matrix θ_{oj} that indicates if it is known that order o fits into bin type j or not. Moreover, in case it is known, ϕ_{oj} indicates if it is feasible or not and Γ the set of infeasible assignments in the current integer solution. If orders are assigned to bins that have not been checked before (line 4), the respective bin packing problem is solved and this assignment is marked as known. If the bin packing problem is infeasible, we add cuts to the relaxed master problem that forbid this bin type order assignment not only for the checked stage and scenario but for all scenarios and stages (line 8). In case of at least one infeasible assignment, we try to repair this solution (lines 13–15) by reassigning orders to other available bins. For this, we develop different strategies, which are detailed in the following section.

Remark 2. While the size of the subproblem linearly increases in the number orders, bin types, scenarios, and stages, in the decomposition the number of 3D-BPPs, which have to be solved, does not increase in the number of scenarios and stages because of the matrix θ_{oj}. The number of scenarios and stages only influences the complexity of the relaxed master problem and the repair method.

4.3. Repair method

We develop two new repair strategies. The first repair strategy, called *shift repair* is a simple and fast strategy that just tries to shift infeasible order-to-bin assignments. The second strategy, called *reOpt repair* solves an auxiliary MILP that further considers effects on procurement and inventory costs.

It is important to note that the repairing step significantly differs in the S3D-BSP compared to the deterministic version. Due to the multi-stage structure and the procurement decisions of bin types, the inventory of bin types is considered. This means that reassigning orders impacts the available inventory and the involved costs.

4.3.1. Shift repair

This strategy is executed for each infeasible assignment of order o to bin type j in each stage t and each scenario s. Algorithm 2 identifies the best fitting bin type $\hat{j}$ in the selected portfolio ($\hat{n}_j = 1$) by solving

the 3D-BPP of Section 3.2. If such a bin type is available in the current portfolio, all n_{ojts} infeasible orders are shifted to the newly selected bin type $\hat{j}$. In case no feasible bin type is available in the current portfolio, the solution cannot be repaired and the execution of the algorithm is immediately terminated (line 5). We further improve this strategy by solving the 3D-BPP only once for each order o to bin type j in each repair step since the resulting best fitting bin $\hat{j}$ is the same for all stages and scenarios. Still the shift is executed for all stages and scenarios.

Algorithm 2: ShiftRepair.

```
   Input: Γ, n_ojts, n̂_j
1  n̄_ojts ← n_ojts
2  for (o, j, t, s) ∈ Γ do
3      j̄ ← solve 3D-BPP for order o with bin types n̂_j
4      if j̄ returns infeasible then
5          return false ;                          ▷ Stop repair
6      else
7          n̄_oj̄ts = n̄_oj̄ts + n_ojts
8          n̄_ojts = 0
9      end
10 end
11 return n̄_ojts
```

However, we ignore all effects on inventory p_{jts} and possibly necessary procurement decision adjustments q_{jts}. Thus, we only insert a partial solution n_{ojts} that the solver can use to generate a feasible solution by finding the right procurement and inventory decisions. Note that this is a default setting in callbacks (we use Gurobi). The solver fixes all inserted decision variables and then tries to find feasible values for the remaining decision variables.

4.3.2. ReOpt repair

In this repair strategy, we again first identify the best fitting bin type for each infeasible order-bin type assignment. But instead of ignoring shift effects, we solve the following modified version of the relaxed master problem, denoted as RMP^R, as auxiliary problem to identify the optimal procurement and inventory strategy. We fix the given bin type portfolio $\hat{n}^*_j$ in the relaxed master problem and further include all generated cuts that forbid infeasible assignments. Therefore, let Λ be the set with all known infeasible assignments (o, j).

$$RMR^R : \min \quad (1)$$

$$\text{s.t.} \quad (3), (5) - (13)$$

$$\hat{q}_{jts} \le \hat{n}^*_j \qquad \forall j \in J, s \in S, t \in T \tag{36}$$

$$n_{ojts} = 0 \qquad \forall (o, j) \in \Lambda, s \in S, t \in T \tag{37}$$

Compared to the previous strategy this implies not only inventory and procurement decision adjustments but also the reassignment of orders to different bin types among the available bin type portfolio. Algorithm 3 details the procedure. This strategy obviously comes with much higher computational effort compared to the ShiftRepair due to line 6. But, it has the potential to generate better solutions. Therefore, we use a parameter α that states that this strategy is only executed every α iterations.

4.4. Improvement strategies

Fontaine and Minner (2023) introduced several acceleration strategies: They evaluated single-item order feasibility and trivial infeasible bin type assignments to add those cuts in the preprocessing. Further, they introduced the concept of bin type and order hierarchy including a variable removal strategy that avoids additional infeasible assignments. The core idea is that if an order can be assigned to one specific bin, this order also fits into a larger bin, i.e., the small bin can be placed into the larger bin. Moreover, if it cannot be assigned to a bin, also smaller bins in the portfolio are infeasible.

Algorithm 3: ReOptRepair.

```
   Input: Γ, n_ojts, n̂*_j
1  for (o, j, t, s) ∈ Γ do
2      j̄, Λ ← solve 3D-BPP for order o with bin types n̂*_j
3      if j̄ returns infeasible then
4          return false ;                          ▷ Stop repair
5  end
6  n*_ojts, p*_jts, q*_jts, q̂*_jts ← solve RMP^R(Λ, n̂*_j)
7  return n*_ojts, p*_jts, q*_jts, q̂*_jts
```

Since the number of bin packing problems that need to be evaluated has a significant impact on the performance of the method, we further extend the concept of order and bin type strategies to scenarios and stages. If an order cannot be assigned to a bin type in one scenario and one stage, this is not possible in any stage or scenario. Therefore, all generated cuts of one assignment can be transferred to all stages and scenarios.

Remark 3. The scenario and stage hierarchy implies that the repair integer node procedure scales independently of the number of scenarios and stages if the shift strategy is used. At maximum $|O| \cdot |J|$ 3D-BPPs need to be solved.

Only the relaxed master problem increases. This holds for both the general relaxed master problem RMR^R and the adjusted model in the ReOpt repair strategy.

Finally, we evaluate the potential of repairing solutions (line 9). For doing so, consider a solution with a selected bin type portfolio $\tilde{J} = \{1, \dots, m\}$, and a selected bin type $\tilde{j}_o$ of order o with $n_{ojts} > 0$. Further let $V(j)$ be the volume of bin type j.

Lemma 1. *If there exists exactly one infeasible assignment $n_{\bar{o}\tilde{j}_{\bar{o}}ts} > 0$, then a lower bound of the volume increase of the repaired solution is given by*

$$n_{\bar{o}\tilde{j}_{\bar{o}}ts}\left(V(\tilde{j}_{\bar{o}}) - min_{j \in \tilde{J}}\left\{V(j) | j \neq \tilde{j}_{\bar{o}} \wedge V(j) > V(\tilde{j}_{\bar{o}})\right\}\right)$$

In case of several infeasible assignments, Lemma 1 is repeatedly applied to all infeasible assignments and the repair strategy is only executed if the costs of the minimal volume increase result in total costs below the costs of the current best found solution.

5. Numerical study

In the numerical study, we analyze the performance of branch-and-repair and give managerial implications for the S3D-BSP. Branch-and-repair is implemented in C++ and we use the Gurobi API (version 11.0) to solve the MILPs. All experiments are conducted on an AMD Ryzen 9 3950X 16-Core Processor, 3.493 GHz with 128 Gb RAM. All instances and results are available at https://github.com/FontainePirmin/S3DBSP.

5.1. Datasets and numerical setup

We build on the datasets introduced by Fontaine and Minner (2023) for the deterministic single-period version of the problem setting and extend those for the S3D-BSP. Explicitly, we use the O-datasets with 100, 200, 300, 400, and 500 orders where each deterministic size consists of five different order settings each (25 instances in total). Each order comprises between one and six items. As a result, the 3D-BPPs exhibit moderate complexity, and the computational effort required to solve the subproblems remains manageable. Based on these instances, we generate a variety of multi-stage stochastic versions with $|T| \in \{4, 5, 6, 7, 8\}$ stages and 2 or 3 scenarios starting at each node in the scenario tree, resulting in the following scenario settings

Table 1
Performance analysis depending repair configuration.

mode	avg time [in sec]	impr. [in %]	opt	avg gap [in %]
no repair	6927	-	4 / 5	18.74
only Shift repair	5035	27.32	4 / 5	0.27
$\alpha = 5$	5544	19.97	4 / 5	0.27
$\alpha = 4$	5221	24.62	4 / 5	0.27
$\alpha = 3$	5468	21.06	4 / 5	0.27
$\alpha = 2$	4029	41.83	5 / 5	0.00
always ReOpt repair	5466	21.08	5 / 5	0.00

$(4, 16), (5, 32), (6, 64), (7, 128), (8, 256), (4, 81), (5, 243)$ where the first number indicates the number of stages and the second number the number of scenarios. For each of these settings, we randomly sample the demand of each order for every stage and scenario from a uniform distribution between 70 and 130 units. For the cost factors, we assume inventory holding costs $c^I = 0.1$, procurement costs $c^O = 5000$, costs of unused space $c^S = 0.001$, and costs of variety $c^V = 300$. Please note that the latter two factors are based on Fontaine and Minner (2023) and have been extended to the new problem setting based on pretest to reflect a dynamic procurement policy.

5.2. Parameter tuning and impact of repair strategies

To find the best performance of branch-and-repair, we analyze the impact of different repair strategies of the algorithm. For this, we used the instances with 100 demands, 6 stages and 2 scenarios per stage. Explicitly, we execute branch-and-repair without repair strategy, only with shift repair, with shift repair and reOpt repair for $\alpha = 2, 3, 4, 5$ (which indicates how often reOpt repair is executed; see Section 4.3) and with reOpt repair executed in each iteration. Besides the average run time in seconds and the number of instances solved to optimality, Table 1 reports the percentage improvement compared to a basic version without repair function.

The analysis shows that the best average run times are achieved when ReOpt repair is executed every second time. Compared to no repair strategy, this leads to an average run time reduction of 41.83 percent. Reducing the frequency or executing ReOpt repair every iteration significantly increases the run time and not all instances can be solved to optimality within the time limit. Moreover, it is worth mentioning that always executing the ReOpt repair strategy finds the optimal solutions but with higher run times. This shows that ReOpt repair (i.e., an advanced repair strategy) is necessary to always find optimal solutions but a too extensive usage comes at costs of run time.

5.3. Performance analysis

In this section, we analyze the performance of branch-and-repair by increasing the problem size along different dimensions. All experiments were executed with the best configuration found in Section 5.2 ($\alpha = 2$), a run time limit of four hours and a gap of one percent as stopping criterion. Such a gap is a widely accepted criterion in two-stage stochastic programs (Birge & Louveaux, 2011).

In Table 2, we analyze the impact of the number of scenarios and stages. Therefore, we use the instances with 100 orders per stage and scenario and vary the number of stages and the number of scenarios that start in each node. We report the average run time in seconds, the average gap in percent, and the number of instances solved to optimality. Please note that due to the stopping criterion, instances solved to optimality include all instances where the algorithm stopped before the time limit.

In the results, we see that branch-and-repair scales very well in the size of the scenario tree. Only if more than 100 scenarios are considered, we cannot prove optimality for all instances. Particularly, in the largest set with 8 stages and 256 scenarios only one of the instances converges. However, the gap remains at 3.15% reasonable. This run time increase is driven by the increasing complexity of the relaxed master problem. However, due to the scenario hierarchy, the number of bin packing problems that need to be solved remains the same. That demonstrates that the method also scales very well within the scenario tree. For example, the relaxed master problem of the largest instance $(8, 256)$ consists of 46,080 binary variables, 91,980 continuous variables, and approximately 4.5 million integer variables that can be relaxed.

Table 2
Performance analysis depending on the number of stages and scenarios.

$\|T\|$	$\|S\|$	avg time [in sec]	opt	avg gap [in %]
4	16	172	5 / 5	0.00
	81	4779	5 / 5	0.00
5	32	1308	5 / 5	0.00
	243	10,749	2 / 5	1.73
6	64	4029	5 / 5	0.00
7	128	8838	4 / 5	0.48
8	256	13,439	1 / 5	3.15

Table 3
Performance analysis depending on the number of orders.

$\|O\|$	avg time [in sec]	opt	avg gap [in %]
100	1308	5 / 5	0.000
200	8966	4 / 5	0.395
300	14,255	1 / 5	1.006
400	13,792	1 / 5	3.304
500	14,400	0 / 5	16.579

In Table 3, we use the instances with five stages and 32 scenarios (2 starting in each node) and increase the number of orders. Again, we report both the average run time in seconds, the number of instances solved to optimality, and the average gap in percent.

The results show that the run time increases with the number of orders. This run time increase is driven by two factors. First, the number of bin packing problems that need to be solved and the size of the relaxed master problem. However, the only decision variable that depends on the number of orders is n_{ojts}, which can be relaxed (Remark 1). The gaps show that for larger sizes, the run time limit of four hours is too low since, particularly with 500 orders, no instance is solved to optimality anymore and the gaps increase. But again, we are considering here very large combinatorial optimization problems and a tactical planning problem. The effectiveness of branch-and-repair can be largely attributed to a key factor: evaluating one bin type portfolio requires solving a very large number of bin packing problems. By decomposing the problem into a relaxed master problem and individual bin packing subproblems, the hierarchies reduce the number of bin packing evaluations. This indicates that branch-and-repair is particularly advantageous when strategic or tactical design decisions are coupled with complex operational combinatorial optimization problems.

6. Case study

To derive managerial implications, we use the real-world data sets from the MSOM data-driven challenge (Guo et al., 2024). It provides the demand of several distribution centers over a year. To generate the scenarios, we used the demand of one week for one scenario of one stage, allowing us to generate four demand periods (five stages) in the multi-stage stochastic program. This reflects also the size of other multi-stage stochastic programming applications in the literature (see, e.g., Körpeoğlu et al., 2011; Zanjani et al., 2010). In Table 4, we give further insights into the demand distribution and the order details and volumes of the different distribution centers. We consider packages with dimensions from 10 to 140 cm length and overall 140 different package sizes. Finally, we removed orders from the data set that had no dimensions and very bulky orders that exceed the largest package size. Such items are

Table 4
Demand details for the case study.

distribution center	$\|O\|$	items			volume			demand per scenario	
		min	max	avg	min	max	avg	avg	std dev
RRSZX040	642	1	23	1.70	0.72	1333.36	393.73	3.89	17.52
RRSZX023	744	1	6	1.39	0.72	1263.60	372.93	1.31	6.90
RRSZX034	922	1	4	1.33	30.10	1019.44	354.41	2.10	9.92
RRSZX061	937	1	4	1.29	30.10	1237.54	354.05	2.45	21.64
RRSZX028	959	1	4	1.31	27.36	1237.54	369.95	1.86	8.89
RRSZX033	1384	1	14	1.40	0.72	1263.60	378.17	2.56	14.50
RRSZX002	1503	1	12	1.42	0.11	1365.82	376.22	2.09	11.98
RRSZX004	1695	1	14	1.40	0.11	1313.40	386.46	2.73	15.51
RRSZX060	1904	1	10	1.47	0.72	1263.60	391.96	4.96	33.74
RRSZX045	1979	1	8	1.46	0.11	1263.60	391.93	3.70	31.59

Note: the values of items, volume, and demand per scenario are shown per order

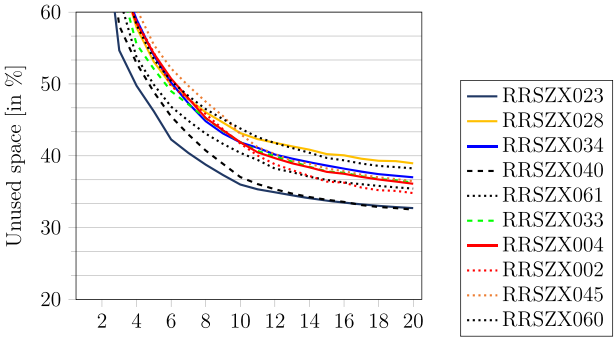

Fig. 1. Unused space depending on portfolio size. .

often also shipped without outer package. While the items considered in the different distribution centers do not vary too much, the difference in the number of orders, max items per order, and the demand (seen in the average demand per scenario and order) and the demand variability is larger.

Similar to Fontaine and Minner (2023), we reduce the size of the set O by aggregating orders with identical item sizes. That is, multiple orders containing the same-sized items are treated as a single representative order $o \in O$, with the corresponding demand reflecting the total number of original orders. As a result, the size of O represents the number of distinct bin packing problems in the warehouse. This reduction is important not only from a computational perspective, as it simplifies the problem space, but also from a managerial standpoint, as it reflects the diversity of order types encountered in operations.

6.1. Effect of portfolio size

To analyze the effect of a regulation that tries to achieve unused space of at maximum 40 percent, we solve the S3D-BSP with predefined portfolio sizes of one to 20.

Fig. 1 shows that the demand significantly influences the optimal portfolio. While for RRSZX023 already eight parcel types satisfy the 40 percent regulation, for RRSZX028 17 parcel types are necessary. This is particularly interesting since the demand details of RRSZX028 (see Table 3) show that this distribution center is in none of the KPIs extreme. This underlines the need of using decision-support models for deriving the optimal portfolio.

While the size of the portfolios varies, the general trend is similar: In the beginning, increasing a portfolio significantly reduces unused space, and then the marginal contribution reduces. Still, these effects are different since several lines cross each other. Moreover, the selected parcel types differ such that, e.g., the eight parcel types are not a subset of the other optimal portfolios. Thus, an individual analysis is necessary.

Result 1. *With increasing portfolio size, the marginal contribution to reduce the unused space decreases.*

6.2. Importance of multi-stage setting

Additionally, we analyze the effect of solving a multi-stage stochastic program. For this, we consider four scenarios:

4-stages: We solve the S3D-BSP and find the minimal portfolio that satisfies the 40-percent unused space regulation.

Full portfolio: We solve the S3D-BSP where all 140 parcels are used. This scenario gives a lower bound on the minimal unused space.

1-stage: We solve a single period variant of the S3D-BSP where only the demand of the first stage is considered. We find the minimal portfolio that satisfies the 40-percent unused space regulation. This scenario illustrates if only the information of the next period is considered.

1-stage (avg): We solve a single period variant of the S3D-BSP where the average demand of 4-stage scenario tree is considered. Again, we find the minimal portfolio that satisfies the 40-percent unused space regulation. This scenario uses the full information but does not solve the more complex multi-stage problem.

For a fair comparison, the derived portfolios are always evaluated in the multi-stage stochastic program. This means that the multi-stage stochastic program was solved with the fixed bin type portfolio of the solution of the respective scenario. Fig. 2 shows the results for each of the distribution center. The dark gray bars show the unused space of the full portfolio and the light gray, blue, and green bars the additional unused space for the other three scenarios. Moreover, the resulting portfolio size is shown on top of each bar. Since the 1-stage scenario might choose a portfolio that does not allow to pack all demands in the multi-stage setting, infeasible portfolios are shown in red.

From the results, we can conclude several insights.

Result 2. *The 1-stage scenario results often in infeasible solutions.*

The 1-stage scenario does not reflect the full demand variety and therefore leads to several infeasible portfolios (7 out of 10). Even though only 2.6 of the orders are effected on average, this shows that the full demand variety needs to be taken into account. For the 2-stage (avg) scenario, this effect may arise when certain orders are not reflected by the average demand. Similarly, in out-of-sample settings with different order structures, such discrepancies can occur as well.

Result 3. *Only the 4-stage scenario that considers the introduced S3D-BSP guarantees the 40-percent target.*

Several evaluated portfolios do not achieve the 40-percent target. This can happen using the 2-stage scenario approach and the 2-stage (avg) scenario approach. Particularly, RRSZX061 shows a very high violation since a relatively small portfolio of eight parcel types is suggested.

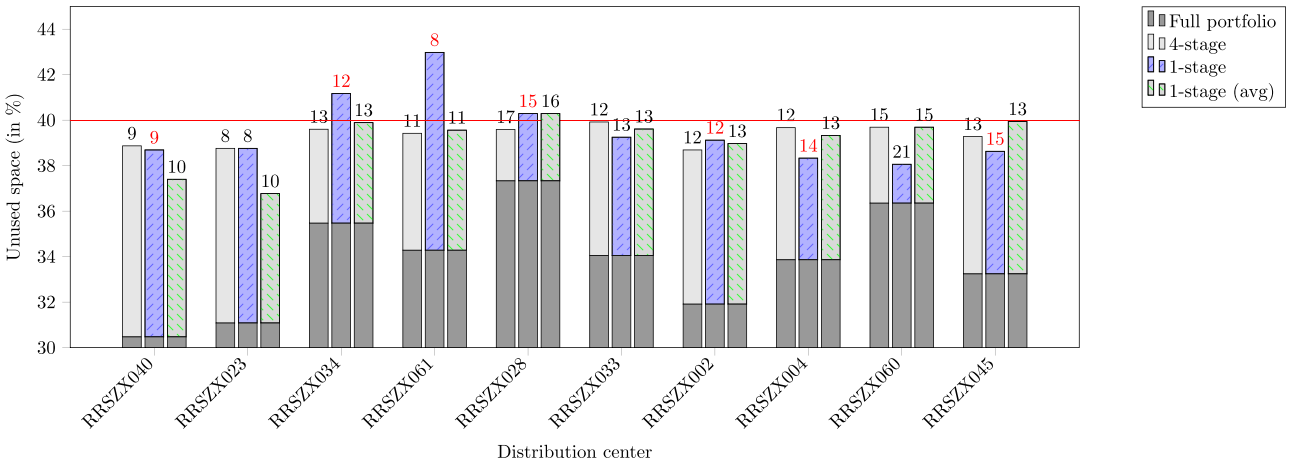

Fig. 2. Unused space (y-axis) and portfolio size (number on top of the bar) to achieve the 40% regulation .
Note: Red portfolio size indicates that the portfolio is infeasible when evaluated in the 4-stage scenario

Table 5
Out-of-sample performance.

distribution center	unused space [in %]		rel dev [in %]
	in-sample	out-of-sample	
RRSZX040	38.87	39.80	2.39
RRSZX023	38.76	38.40	−0.93
RRSZX034	39.60	39.98	0.96
RRSZX061	39.42	39.52	0.25
RRSZX028	39.59	39.82	0.58
RRSZX033	39.92	40.93	2.53
RRSZX002	38.69	38.87	0.47
RRSZX004	39.67	39.65	−0.05
RRSZX060	39.69	40.18	1.23
RRSZX045	39.28	39.55	0.69
avg	39.35	39.67	0.82

Result 4. *Both 1-stage scenarios can result in significantly smaller and larger portfolio sizes. Moreover, there is no connection between portfolio size and the demand.*

The optimal portfolio sizes vary. For example, RRSZX023 has an optimal portfolio of eight, while the average scenario suggests a portfolio of ten. This results then in less unused space but was not the target of the optimization. Contrary, for RRSZX0028, the portfolio is smaller, resulting in the already discussed violation. But even if the same-sized portfolios are selected (RRSZX045), these portfolios differ and increase unused space. This effect is also shown for RRSZX002 where a large portfolio leads to more unused space.

This analysis has shown that both the portfolio size and the portfolio itself can significantly differ if the multi-stage setting is ignored. Moreover, the resulting effects on unused space are not predictable. Therefore, a multi-stage setting should be solved.

6.3. Out-of-sample evaluation

Until now, we have shown that the S3D-BSP performs well in-sample. In this section, we evaluate its out-of-sample performance. For this purpose, we generate for each distribution center 16 new out-of-sample scenarios by sampling each order's demand based on its historical demand using a normal distribution. Each scenario is then evaluated using the optimal portfolio derived in the previous section. Table 5 shows the in-sample performance, the out-of-sample performance, and the corresponding percentage deviation.

Building on the in-sample performance of the previous section, we see that the out-of-sample performance remains good. On average, the out-of-sample utilization is 0.82 percent (0.32 percentage points) higher than the in-sample utilization. Two warehouses did not meet the 40-percent target. Among all warehouses, RRSZX033 stands out with an out-of-sample utilization of only 40.93 percent. However, this result is consistent with its in-sample performance, which was already close to the threshold at 39.92 percent.

Result 5. *On average, the out-of-sample performance is good. In practice, decision-makers may consider including a small buffer when planning to ensure that utilization targets are also met under out-of-sample conditions.*

7. Conclusion

We introduced the S3D-BSP to address the design of a parcel type portfolio under demand uncertainty and formulated it as a multi-stage stochastic program. To solve large instances, we developed a branch-and-repair method and showed how balancing different repair strategies can improve the run time. It is not beneficial always aiming for high-quality solutions in the repair step. The numerical results further showed that our method is capable of solving multi-stage stochastic programs efficiently with combinatorially challenging subproblems. The proposed method is highly adaptable and can be used with many variants of packing problems and could even use heuristics for solving packing problems, resulting in a matheuristic. This flexibility is particularly relevant from a practical perspective, as in real-world applications, complex packing problems would typically also be solved using heuristics. Moreover, using constraint programming versions of the 3D-BPP (see, e.g., Martin et al., 2024) would allow to include constraint programming into branch-and-repair. From a managerial perspective, we have seen that the portfolio and its size largely varies depending on the demand. Ignoring the multi-stage setting can lead to infeasible portfolios or not satisfying the 40-percent regulation. Therefore, it is important to use such a multi-stage problem, particularly since branch-and-repair allows to scale to multiple stages.

Despite the limited number of stages, the out-of-sample performance was satisfactory. This result may be influenced by the underlying demand and problem structure. Future studies in other industries should investigate the robustness of the proposed approach. Moreover, such additional analyses using diverse real-world data sets would be valuable to further validate the findings.

Methodologically, it would be interesting to apply branch-and-repair to other multi-stage stochastic problem settings both with similar and with different structures to demonstrate its generalizability. So far, branch-and-repair has mainly been applied to problem settings in which the subproblems are relatively complex, allowing for substantial computational savings through the decomposition and hierarchies.

CRediT authorship contribution statement

Pirmin Fontaine: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

I declare that there are no conflicts of interest regarding the publication of this paper. I confirm that I have no relevant financial or non-financial interests to disclose.

Acknowledgments

I thank the editor and anonymous reviewers whose constructive comments helped to significantly improve the paper.

References

Ali, S., Ramos, A. G., Carravilla, M. A., & Oliveira, J. F. (2022). On-line three-dimensional packing problems: A review of off-line and on-line solution approaches. *Computers & Industrial Engineering*, *168*, 108122.

Alonso, M. T., Alvarez-Valdes, R., Parreño, F., & Tamarit, J. M. (2016). Determining the best shipper sizes for sending products to customers. *International Transactions in Operational Research*, *23*(1–2), 265–285.

Alptekinoğlu, A., Banerjee, A., Paul, A., & Jain, N. (2013). Inventory pooling to deliver differentiated service. *Manufacturing & Service Operations Management*, *15*(1), 33–44.

Axsäter, S. (2015). Multi-echelon systems: Lot sizing. In *Inventory Control* , pp. 171–189. Cham: Springer International Publishing.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, *4*(1), 238–252.

Bertazzi, L., & Maggioni, F. (2018). A stochastic multi-stage fixed charge transportation problem: Worst-case analysis of the rolling horizon approach. *European Journal of Operational Research*, *267*(2), 555–569.

Biggs, K. (2021). Sustainability in e-commerce: How can online retailers make their business greener? https://parcellab.com/e-commerce/sustainability-online-retail/.

Birge, J. R. (1985). Decomposition and partitioning methods for multistage stochastic linear programs. *Operations Research*, *33*(5), 989–1007.

Birge, J. R., & Louveaux, F. (2011). Introduction to stochastic programming. New York, NY: Springer New York.

Brinker, J., & Gündüz, H. I. (2016). Optimization of demand-related packaging sizes using a p-median approach. *The International Journal of Advanced Manufacturing Technology*, *87*(5), 2259–2268.

Brylla, D., & Walsh, G. (2022). When faster online delivery backfires: Examining the negative consequences of split deliveries. *International Journal of Electronic Commerce*, *26*(4), 497–525. https://doi.org/10.1080/10864415.2022.2123647

Castro, P. M., & Oliveira, J. F. (2011). Scheduling inspired models for two-dimensional packing problems. *European Journal of Operational Research*, *215*(1), 45–56.

Çelik, Ş., Martin, L., Schrotenboer, A. H., & Van Woensel, T. (2025). Exact two-step benders decomposition for the time window assignment traveling salesperson problem. *Transportation Science*, *59*(2), 210–228.

Chen, C. S., Lee, S. M., & Shen, Q. S. (1995). An analytical model for the container loading problem. *European Journal of Operational Research*, *80*(1), 68–76.

Crainic, T. G., & Montreuil, B. (2016). Physical internet enabled hyperconnected city logistics. *Transportation Research Procedia*, *12*, 383–398.

Crainic, T. G., Ricciardi, N., & Storchi, G. (2009). Models for evaluating and planning city logistics systems. *Transportation Science*, *43*(4), 432–454.

Côté, J.-F., Dell'Amico, M., & Iori, M. (2014). Combinatorial Benders' cuts for the strip packing problem. *Operations Research*, *62*(3), 643–661.

Côté, J.-F., Haouari, M., & Iori, M. (2021). Combinatorial Benders decomposition for the two-dimensional bin packing problem. *INFORMS Journal on Computing*, *33*(3), 963–978.

Daryalal, M., Bodur, M., & Luedtke, J. R. (2022). Lagrangian dual decision rules for multistage stochastic mixed-integer programming. *Operations Research*, *72*(2), 717–737.

Delorme, M., Iori, M., & Martello, S. (2016). Bin packing and cutting stock problems: Mathematical models and exact algorithms. *European Journal of Operational Research*, *255*(1), 1–20.

DHL (2022). Artificial intelligence saves costs and emissions by optimizing packaging of shipments for DHL supply chain customers. https://www.dpdhl.com/en/media-relations/press-releases/2022/artificial-intelligence-optimizing-packaging-for-dhl-supply-chain-customers.html.

Dowsland, K. A., Soubeiga, E., & Burke, E. (2007). A simulated annealing based hyperheuristic for determining shipper sizes for storage and transportation. *European Journal of Operational Research*, *179*(3), 759–774.

Esser, K., & Kurte, J. (2020). KEP-studie 2019 - analyse des marktes in deutschland (CEP study 2019 - Analysis of the German market). Technical Report BIEK.

European Union (2024). Reducing packaging waste - review of rules. https://ec.europa.eu/info/law/better-regulation/have-your-say/initiatives/12263-Reducing-packaging-waste-review-of-rules_en.

Fontaine, P., & Minner, S. (2023). A branch-and-repair method for three-dimensional bin selection and packing in e-commerce. *Operations Research*, *71*(1), 273–288.

Friesen, D. K., & Langston, M. A. (1986). Variable sized bin packing. *SIAM Journal on Computing*, *15*(1), 222–230.

Guo, X., Yu, Y., Allon, G., Wang, M., & Zhang, Z. (2024). Rirshun logistics: Home appliance delivery data for the 2021 manufacturing & service operations management data-driven research challenge. *Manufacturing & Service Operations Management*, *26*(4), 1358–1371.

Gzara, F., Elhedhli, S., & Yildiz, B. C. (2020). The pallet loading problem: Three-dimensional bin packing with practical constraints. *European Journal of Operational Research*, *287*(3), 1062–1074.

Hooker, J. N. (2007). Planning and scheduling by logic-based Benders decomposition. *Operations Research*, *55*(3), 588–602.

Hooker, J. N., & Ottosson, G. (2003). Logic-based Benders decomposition. *Mathematical Programming*, *96*(1), 33–60.

Forbes Insights and D.S. Smith (2018). The empty space economy. Technical Report Forbes Insights and DS Smith.

Iori, M., de Lima, V. L., Martello, S., Miyazawa, F. K., & Monaci, M. (2021). Exact solution techniques for two-dimensional cutting and packing. *European Journal of Operational Research*, *289*(2), 399–415.

Johansen, B. D. (2024). How to get the parcel mix right for your distribution centre? https://www.beumergroup.com/knowledge/cep/parcel-sizes-mix-distribution/.

Kall, P., & Wallace, S. W. (1994). Stochastic programming. Chichester: Wiley.

Kayacık, S. E., Basciftci, B., Schrotenboer, A. H., & Ursavas, E. (2025). Partially adaptive multistage stochastic programming. *European Journal of Operational Research*, *321*(1), 192–207.

Keutchayan, J., Munger, D., & Gendreau, M. (2020). On the scenario-tree optimal-value error for stochastic programming problems. *Mathematics of Operations Research*, *45*(4), 1572–1595.

Körpeoğlu, E., Yaman, H., & Selim Aktürk, M. (2011). A multi-stage stochastic programming approach in master production scheduling. *European Journal of Operational Research*, *213*(1), 166–179.

Laporte, G., & Louveaux, F. V. (1993). The integer l-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, *13*(3), 133–142.

Laporte, G., Louveaux, F. V., & Van Hamme, L. (2002). An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, *50*(3), 415–423.

Martello, S., Pisinger, D., & Vigo, D. (2000). The three-dimensional bin packing problem. *Operations Research*, *48*(2), 256–267.

Martin, M., de Queiroz, T. A., & Morabito, R. (2024). Solving the three-dimensional open-dimension rectangular packing problem: A constraint programming model. *Computers & Operations Research*, *167*, 106651.

Montreuil, B., Ballot, E., & Tremblay, W. (2015). Modular design of physical internet transport, handling and packaging containers. In *Progress in material handling research: 2014*. MHI (*vol. 13*). International Material Handling Research Colloquium.

Nascimento, O. X. d., Alves de Queiroz, T., & Junqueira, L. (2021). Practical constraints in the container loading problem: Comprehensive formulations and exact algorithm. *Computers & Operations Research*, *128*, 105186.

Paquay, C., Schyns, M., & Limbourg, S. (2016). A mixed integer programming formulation for the three-dimensional bin packing problem deriving from an air cargo application. *International Transactions in Operational Research*, *23*(1–2), 187–213.

Pitney Bowes (2023). Parcel shipping index 2023. Technical Report Pitney Bowes.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, *259*(3), 801–817.

Salma, M., & Ahmed, F. (2011). Three-dimensional bin packing problem with variable bin length application in industrial storage problem. In *2011 4th international conference on logistics* (pp. 508–513). IEEE.

šafak, Ö., & Erdoğan, G. (2023). A large neighbourhood search algorithm for solving container loading problems. *Computers & Operations Research*, *154*, 106199.

Thorsteinsson, E. S. (2001). Branch-and-check: A hybrid framework integrating mixed integer programming and constraint logic programming. In *International conference on principles and practice of constraint programming* (pp. 16–30). Springer.

Tsao, Y.-C., Tai, J.-Y., Vu, T.-L., & Chen, T.-H. (2024). Multiple bin-size bin packing problem considering incompatible product categories. *Expert Systems with Applications*, *247*, 123340.

UPS (2021). Cube optimization. https://www.ups.com/media/en/CubeOptimizationSalesSheet.pdf.

Van Slyke, R. M., & Wets, R. (1969). L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, *17*(4), 638–663.

Vieira, M. V. C., Ferreira, F., Duque, J. C. M., & Almeida, R. M. P. (2021). On the packing process in a shoe manufacturer. *Journal of the Operational Research Society*, *72*(4), 853–864.

Wäscher, G., Haußner, H., & Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, *183*(3), 1109–1130.

Zanjani, M. K., Nourelfath, M., & Ait-Kadi, D. (2010). A multi-stage stochastic programming approach for production planning with uncertainty in the quality of raw materials and demand. *International Journal of Production Research*, *48*(16), 4701–4723.

Zhang, X., Chen, L., Gendreau, M., & Langevin, A. (2022). A branch-and-price-and-cut algorithm for the vehicle routing problem with two-dimensional loading constraints. *Transportation Science*, *56*(6), 1618–1635.

Zhang, Y., Sun, L., Hu, X., & Zhao, C. (2019). Order consolidation for the last-mile split delivery in online retailing. *Transportation Research Part E: Logistics and Transportation Review*, *122*, 309–327.