The discrete forward-reserve problem – Allocating space, selecting products, and area sizing in forward order picking

Rico Walter, Nils Boysen and Armin Scholl
02/2012

Working Papers in Supply Chain Management
Friedrich-Schiller-University of Jena
The discrete forward-reserve problem –
Allocating space, selecting products, and
area sizing in forward order picking

Rico Walter\textsuperscript{a}, Nils Boysen\textsuperscript{b}, Armin Scholl\textsuperscript{a}

\textsuperscript{a}Friedrich-Schiller-Universität Jena, Lehrstuhl für Management Science,
Carl-Zeiß-Straße 3, D-07743 Jena, Germany,
{rico.walter,armin.scholl}@uni-jena.de

\textsuperscript{b}Friedrich-Schiller-Universität Jena, Lehrstuhl für Operations Management,
Carl-Zeiß-Straße 3, D-07743 Jena, Germany, nils.boysen@uni-jena.de

*Corresponding author, phone +49 3641 943-100.

Abstract
To reduce labor-intensive and costly order picking activities, many distribution centers are subdivided into a forward area and a reserve (or bulk) area. The former is a small area where most popular stock keeping units (SKUs) can conveniently be picked, and the latter is applied for replenishing the forward area and storing SKUs that are not assigned to the forward area at all. Clearly, reducing SKUs stored in forward area enables a more compact forward area (with reduced picking effort) but requires a more frequent replenishment. To tackle this basic trade-off, different versions of forward-reserve problems determine the SKUs to be stored in forward area, the space allocated to each SKU, and the overall size of the forward area. As previous research mainly focuses on simplified problem versions, where the forward area can continuously be subdivided, we investigate discrete forward-reserve problems. Important subproblems are defined and efficient solution procedures are introduced and tested.

Keywords: Logistics; Warehousing; Order picking; Forward-reserve problem

1 Introduction
Picking highly-demanded stock keeping units (SKUs) directly from bulk storage typically requires removal from deep-lane pallet racks and causes unproductive travel between far-
distant picking locations. Thus, especially in high-volume distribution, many warehouses are subdivided into a forward and a reserve area. The forward area serves as a “warehouse within a warehouse” and stores popular SKUs in easy-to-access racks, e.g., gravity flow racks, concentrated in a compact area. The use of a forward area improves order picking efficiency, but requires additional effort for replenishing the forward area from the reserve area (double handling). Clearly, reducing SKUs stored in forward area enables a more compact forward area (with reduced picking effort) but requires a more frequent replenishment. To tackle this basic trade-off, forward-reserve problems have been formulated and modeled to determine the SKUs to be stored in the forward area, the space allocated to each SKU, and the overall size of the forward area.

Hackman and Rosenblatt (1990) were the first that formulated a mathematical model to allocate space in a continuously divisible forward area and proposed a greedy heuristic. Further contributions stem from Frazelle et al. (1994), who extended the model by regarding the size of a forward area as a decision variable, Van den Berg et al. (1998), who optimized unit-load replenishments that take place during busy and idle periods, Bartholdi and Hackman (2008), who analyzed two wide-spread real-world stocking strategies for small parts in a forward area, and Gu et al. (2010), who provided a branch-and-bound algorithm for solving the joint assignment and allocation problem. Except for the contribution of Van den Berg et al. (1998), all these studies presuppose the “fluid model”, where a forward area can continuously be partitioned among SKUs. Clearly, this simplifying assumption might be justified if merely an approximate benchmark solution is sought. However, for a detailed stocking plan of the forward area, the fluid model shows some severe drawbacks (Bartholdi and Hackman, 2011):

- SKUs can only be stored in discrete units, so that a continuous distribution of space in either case can only be an approximation of reality. This approximation becomes the less accurate the larger an item compared to the size of the shelves.

- Often the number of units stocked in forward area cannot be increased piece-wise but only in steps of multiple units. For instance, consider a gravity flow rack where each lane is filled with units of a single homogeneous product. In this case, e.g., either 0, 1000 or 2000 units can be stored depending on whether none, one or two lanes (each having a capacity for 1000 units) are allocated to the respective SKU.

- Finally, some SKUs show sub-additivity with regard to space because they can be stored in a nested fashion. Consider bathtubs where two units occupy only a little more space than a single unit. This coherency cannot be considered within the continuous forward-reserve problem, because here a linear increase of space utilization is presupposed.

To avoid these shortcomings, this paper considers the discrete forward-reserve problem. Specifically, we investigate three different problem settings each within two important parameter constellations, so that in total six problem cases are treated. The most basic problem setting is the discrete forward-reserve allocation problem (DFRAP), where the given space of a forward area is to be partitioned among a predetermined set of SKUs (see
Sect. 2). The discrete forward-reserve assignment and allocation problem (DFRAAP) combines the space allocation problem with the assignment problem of selecting the products to be stored in the forward area (see Sect. 3). Finally, the discrete forward-reserve allocation and sizing problem (DFRASP) treats the allocation problem jointly with the sizing problem, i.e., for a given set of products a forward area of variable size is to be allocated (see Sect. 4).

With regard to the parameter constellations, we differentiate between variable storage modes and equally sized shelves. In the former case, for any product different storage alternatives exist. For instance, SKUs can be stored in small or large boxes, on one or more pallets in shelves of variable size, or in one or more lanes of varying capacity in a gravity flow rack, where each alternative corresponds to a certain storage mode. Each mode is defined by a specific space utilization and the corresponding discrete number of stored product units. Clearly, the confinement to equally sized shelves defines a special case of variable storage modes. Here, the storage mode of each SKU is already defined and the decision reduces to determining the discrete number of equally sized shelves (each storing an identical multiple of units) to be assigned to any product. Typically, standardized racks with equally sized shelves, e.g., identical lanes in a gravity flow rack, are applied in a wide range of distribution centers as they allow for fast and simple re-assignments, i.e., some products that have been stored forward during the last period will be partly or fully replaced by new products within the next period. Furthermore, standardized racks, typically, cause less investment cost.

In the following, the DFRAP, the DFRAAP, and the DFRASP are treated in Sect. 2, 3, and 4, respectively. In each section both parameter constellations are investigated by defining the respective problem version and either providing a polynomial time solution procedure or proving NP-hardness. Furthermore, in each case we explore the loss in precision of the continuous problem version compared to the discrete case in a comprehensive computational study. Finally, Sect. 5 concludes the paper.

2 The discrete forward-reserve allocation model (DFRAP)

2.1 DFRAP with variable storage modes

Consider a predefined set $P$ of products (or SKUs) to be stored in a forward area of given size $S$, where due to the compact size of the forward area the locations of SKUs are assumed to not affect picking efficiency. Each product $i \in P$ can be stored in one of $n_i$ possible modes $j = 1, \ldots, n_i$ in the forward area. Associated with each mode $j$ for SKU $i$ is a value $a_{ij} > 0$ which gives the (integral) number of units of SKU $i$ that can be stored in mode $j$. The space required by storing product $i$ in mode $j$ is denoted by $w_{ij}$. Throughout the paper we assume that the storage modes of each product $i$ are labeled so that $w_{1i} < w_{2i} < \ldots < w_{ni}$. According to this, we consider only non-dominated storage modes for each SKU $i \in P$, i.e., $a_{i1} < \ldots < a_{in_i}$.

As formulated in the mathematical model (1)-(4), the DFRAP decides on the mode in which each SKU is stored in the forward area (restriction (2)). Equivalently, it decides on the number of units to be stored per product without exceeding given storage space.
S (restriction (3)) which is assumed to be at least as large as \( \sum_{i \in P} w_i \). Otherwise, not all products of the predefined set could be stored in the forward area. Binary variables \( x_{ij} \) defined in (4) indicate which storage mode is chosen for SKU \( i \). Clearly, the more space is associated with each SKU the less restocks are required per time unit. If \( d_i \) represents the total demand for product \( i \) during the planning period, e.g., a year, then the number of restocks per SKU can simply be calculated by dividing the total demand \( d_i \) by the number of units stored in the forward area of the respective product. Note that the underlying assumptions with regard to restocks are discussed in detail by Bartholdi and Hackman (2008).

When weighting the number of restocks with product specific replenishment cost \( c_i \) the objective (1) of the DFRAP is to minimize the overall restock cost per planning period:

\[
\text{DFRAP: Minimize } C_1(x) = \sum_{i \in P} \sum_{j=1}^{n_i} c_i \frac{d_i}{a_{ij}} x_{ij} \tag{1}
\]

subject to

\[
\sum_{j=1}^{n_i} x_{ij} = 1 \quad \forall i \in P \tag{2}
\]

\[
\sum_{i \in P} \sum_{j=1}^{n_i} w_{ij} x_{ij} \leq S \tag{3}
\]

\[
x_{ij} \in \{0, 1\} \quad \forall i \in P; j = 1, \ldots, n_i \tag{4}
\]

The DFRAP is mathematically equivalent to the well-known multiple-choice knapsack problem (MCKP), which becomes obvious when interpreting the parameters of the MCKP as follows:

- SKUs correspond to classes and (storage) modes (of a SKU) correspond to items (of a class),
- the profit of item \( j \) of class \( i \) is \( p_{ij} = -c_i d_i/a_{ij} \) (minimizing \( \sum_i \sum_j p_{ij} x_{ij} \) is equivalent to maximizing \( \sum_i \sum_j -p_{ij} x_{ij} \)),
- the weight of item \( j \) of class \( i \) equals the space \( w_{ij} \) required by storing SKU \( i \) in mode \( j \),
- the capacity of the knapsack is \( c = S \),
- the size of class \( i \) is \( n_i \).

Clearly, the DFRAP is \( \mathcal{NP} \)-hard as the MCKP, which is a generalization of the knapsack problem, is well-known to be \( \mathcal{NP} \)-hard (see Kellerer et al., 2004, Ch. 11). Furthermore, pseudo-polynomial solution procedures for the MCKP exist, e.g., the dynamic programming approach of Dudzinski and Walukiewicz (1987), so that it follows that the DFRAP is \( \mathcal{NP} \)-hard in the ordinary sense.
2.2 DFRAP with equally sized shelves

For adopting the basic DFRAP to the case of equally sized shelves of a rack or lanes of identical capacity in a gravity flow rack, we keep to the notation introduced in Sect. 2.1 with a slightly different interpretation of $S, w_{ij},$ and $a_{ij}$. Parameter $S$ denotes the total (integral) number of shelves that are available for storing the products of set $P$ in the forward area. Hence, we let $w_{ij}$ denote the number of shelves that are available when storing SKU $i$ in mode $j$. As we agreed on allowing any integral number of shelves between 1 and $n_i$ for forward storing of SKU $i$, where $n_i = S - |P| + 1$ if no other upper bound on the maximum number of shelves is given, we have $w_{ij} = j$ for all $i \in P$ and $j = 1, \ldots, n_i$. Finally, we let $a_i$ denote the number of units of SKU $i$ that can be stored in one shelf, which means that $a_{ij} = j \cdot a_i$ for all $i \in P$ and $j = 1, \ldots, n_i$. Using this modified notation, the DFRAP with equal shelves (denoted as DFRAP$\text{ES}$) can be rewritten as follows:

\[
\text{DFRAP}'\text{ES}: \text{Minimize } C'_{\text{ES}1}(x) = \sum_{i \in P} \sum_{j=1}^{n_i} c_i \frac{d_i}{j \cdot a_i} x_{ij} \tag{5}
\]

subject to

\[
\sum_{j=1}^{n_i} x_{ij} = 1 \quad \forall i \in P \tag{6}
\]

\[
\sum_{i \in P} \sum_{j=1}^{n_i} j \cdot x_{ij} \leq S \tag{7}
\]

\[
x_{ij} \in \{0, 1\} \quad \forall i \in P; j = 1, \ldots, n_i \tag{8}
\]

To shorten notation we define $q_i := c_i d_i / a_i$ as the aggregate restock costs of SKU $i$ for the planning period, if only a single shelf is allocated to $i$. Thus, $q_i/j$ denotes the aggregate restock costs of SKU $i$ when $j$ shelves are allocated to $i$.

In order to avoid infeasible or trivial solutions, we assume $|P| \leq S < \sum_{i \in P} n_i$. Hence, any optimal solution satisfies constraint (7) with equality. Then, using integer variables $x_i$ for all SKUs $i \in P$ the DFRAP$\text{ES}$ can be rewritten in its compact form as follows:

\[
\text{DFRAP}_\text{ES}: \text{Minimize } C_{\text{ES}1}(x) = \sum_{i \in P} \frac{q_i}{x_i} \tag{9}
\]

subject to

\[
\sum_{i \in P} x_i = S \tag{10}
\]

\[
x_i \in \{1, \ldots, n_i\} \quad \forall i \in P \tag{11}
\]

The compact formulation (9)-(11) – which also arises in family disaggregation at hierarchical production planning (cf. Bitran and Hax, 1977) – reveals the equivalence of the DFRAP$\text{ES}$ and the simple (discrete) resource allocation problem with (generalized) lower
bounds \( x_i \geq 1 \) and upper bounds \( x_i \leq n_i \) for all SKUs \( i \in P \) (cf. Katoh and Ibaraki, 1998). The objective function \( \sum_{i \in P} f_i(x_i) \) where \( f_i(x_i) = q_i / x_i \) is separable and convex as each \( f_i \) is a convex function of one variable. In our case, the \( f_i \)'s are even strictly convex and thus the difference or decrement

\[
d_i(x_i) \equiv f_i(x_i) - f_i(x_i + 1) \quad (x_i \in \{1, \ldots, n_i - 1\})
\]

(12)
is decreasing in \( x_i \) for all \( f_i \).

Several efficient polynomial time algorithms have been developed for the integer resource allocation problem (cf. Ibaraki and Katoh, 1988, Chapter 4). The fastest one is the weakly polynomial time algorithm of Frederickson and Johnson (1982) that has an optimal running time of \( O(\max\{|P|, |P| \log((S-|P|)/|P|)}) \). For another algorithm with the same run time complexity see Hochbaum (1994).

### 2.3 Relation between DFRAP\(_{ES}\) and its continuous counterpart

Relaxing the integrity-condition (11) in DFRAP\(_{ES}\) leads to the continuous relaxation denoted by FMB which is a generalization of the classic fluid model (FM) by Hackman and Rosenblatt (1990):

FMB: Minimize \( FC(z) = \sum_{i \in P} \frac{q_i}{z_i} \)

subject to

\[
\sum_{i \in P} z_i = S
\]

(14)

\[
1 \leq z_i \leq n_i \quad \forall i \in P
\]

(15)

In contrast to FMB, the classic fluid model neither considers additional lower nor upper bounds for variables, i.e., (15) is replaced by \( z_i > 0 \). Solving FM by means of convex nonlinear programming (cf. Rockafellar, 1970) results in an optimal allocation of

\[
z^*_i = \frac{S\sqrt{q_i}}{\sum_{k \in P} \sqrt{q_k}}
\]

(16)

shelves to SKU \( i \) and the corresponding optimal Lagrangian multiplier for constraint (14) is (cf. Hackman and Rosenblatt, 1990)

\[
\lambda^*_{FM} = -\frac{(\sum_{k \in P} \sqrt{q_k})^2}{S^2}.
\]

(17)

The more general problem FMB can be solved, for instance, by application of a recursive algorithm proposed by Bitran and Hax (1981) that repeatedly allocates shelves according to (16) till all SKUs \( i \) received at least one shelf and at most \( n_i \) shelves. One iteration consists in identifying SKUs that received less than one shelf or more than \( n_i \) shelves, either increasing all allocations that are smaller than 1 exactly to this lower
bound or decreasing all allocations larger than \( n_i \) exactly to the respective upper bound, and then reallocating the remaining number of shelves among the remaining SKUs. More precisely, let \( S_t \) and \( P_t \) denote the remaining number of free shelves and the remaining set of SKUs for which the optimal allocations have not yet been determined at the beginning of iteration \( t \), respectively. Then, obtain \( z_i^*(S_t) \) for all \( i \in P_t \) by solving FM restricted to \( S_t \) and \( P_t \). In case \( 1 \leq z_i^*(S_t) \leq n_i \) for all \( i \in P_t \), set \( z_i^* = z_i^*(S_t) \) for all \( i \in P_t \) and stop the procedure as an optimal solution to \( FMB \) has been found. Otherwise, define \( P_t^+ = \{ i : z_i^*(S_t) > n_i \} \) as well as \( P_t^- = \{ i : z_i^*(S_t) < 1 \} \) and compute \( \Delta_t^+ = \sum_{i \in P_t^+} (z_i^*(S_t) - n_i) \) as well as \( \Delta_t^- = \sum_{i \in P_t^-} (1 - z_i^*(S_t)) \). If \( \Delta_t^+ \geq \Delta_t^- \), set \( z_i^* = n_i \) for all \( i \in P_t^+ \), \( P_{t+1} = P_t \setminus P_t^+ \), and \( S_{t+1} = S_t - \sum_{i \in P_t^+} n_i \). Otherwise, set \( z_i^* = 1 \) for all \( i \in P_t^- \), \( P_{t+1} = P_t \setminus P_t^- \) and \( S_{t+1} = S_t - |\Delta_t^-| \).

Since at each iteration the optimal number of shelves is determined for at least one SKU, this procedure requires at most \(|P|\) iterations where iteration \( t \) consumes at most \( O(|P| - t) \) time. Thus, the worst-case running time of Bitran and Hax’s algorithm is \( O(|P|^2) \) time.

Interestingly, the values \( z_i^* \) \((i \in P)\) can also be used to efficiently determine an optimal solution for the discrete problem version (cf. Ibaraki and Katoh, 1988, Sect. 4.6). Therefore, let \( \hat{t} \) denote that iteration of Bitran and Hax’s algorithm at which an optimal \( FMB \)-solution has been found. Then, since \( z^* = (z_1^*, \ldots, z_{|P|}^*) \) is optimal, the Kuhn-Tucker conditions (cf., e.g., Rockafellar, 1970) are satisfied and we receive the optimal Lagrangian multiplier \( \lambda_{FMB}^* = -\frac{(\sum_{i \in P} \sqrt{n_i})^2}{\hat{t}} \) associated with constraint (14) of problem \( FMB \). Furthermore, due to the convexity of the functions \( f_i \) for all \( i \), we can conclude

\[
\lambda_{FMB}^* + d_i(x_i) \geq 0 \quad \text{for all} \quad 1 \leq x_i \leq \lfloor z_i^* \rfloor - 1 \tag{18}
\]

and

\[
\lambda_{FMB}^* + d_i(x_i) \leq 0 \quad \text{for all} \quad \lfloor z_i^* \rfloor + 1 \leq x_i \leq n_i. \tag{19}
\]

Then, by setting

\[
\bar{x}_i = \begin{cases} 
 z_i^* & \text{if } z_i^* \text{ is integral,} \\
 \lfloor z_i^* \rfloor & \text{if } \lambda_{FMB}^* + d_i(\lfloor z_i^* \rfloor) \leq 0, \\
 \lfloor z_i^* \rfloor + 1 & \text{if } \lambda_{FMB}^* + d_i(\lfloor z_i^* \rfloor) > 0,
\end{cases} \tag{20}
\]

for all \( i \in P \) we obtain an optimal solution to the Lagrangian relaxation of \( DFRAP_{ES} \) with the right-hand side \( S \) being replaced by \( \sum_{i \in P} \bar{x}_i \). Verifying the optimality of \( \bar{x} \) is straightforward by equivalently reformulating the objective of the Lagrangian relaxation for a given real number \( \lambda \) as follows:

\[
\min \left( \sum_{i \in P} f_i(x_i) - \lambda \left( \sum_{i \in P} x_i - S \right) \right) \iff \min \sum_{i \in P} \left( f_i(x_i) - \lambda x_i \right) \iff \\
\iff \min \sum_{i \in P} \left( f_i(1) - (d_i(1) + \lambda) - (d_i(2) + \lambda) - \ldots - (d_i(x_i - 1) + \lambda) - \lambda \right) \\
\iff \max \sum_{i \in P} \left( (d_i(1) + \lambda) + (d_i(2) + \lambda) + \ldots + (d_i(x_i - 1) + \lambda) \right).
\]
Since $\tilde{x}$ is optimal, $\tilde{D} = \{d_i(y) \mid i \in P; y = 1, \ldots, \tilde{x}_i - 1\}$ contains the $\tilde{S} - |P|$ largest elements of the set $D = \{d_i(y) \mid i \in P; y = 1, \ldots, n_i - 1\}$ of all decrements where $\tilde{S} = \sum_{i \in P} \tilde{x}_i$. Then, it remains to determine the $(\tilde{S} - \tilde{S})$ largest elements in $D \setminus \tilde{D}$ if $\tilde{S} < S$ or the $(\tilde{S} - S)$ smallest elements in $\tilde{D}$ if $\tilde{S} > S$. Clearly, as $|\sum_{i \in P} \tilde{x}_i - \sum_{i \in P} z^*_i| \leq |P| - 1$, this can be done in $O(|P| \log |P|)$ time (cf. Weinstein and Yu, 1973) or even faster in $O(|P|)$ time (cf. Ibaraki and Katoh, 1988, Sect. 4.6) based on the algorithm of Frederickson and Johnson (1982). Note that the latter observation on $|\sum_{i \in P} \tilde{x}_i - \sum_{i \in P} z^*_i|$ induces that the maximal absolute difference in the discrete and the continuous optimal allocation is bounded above by $|P|$ for each SKU $i$, more formally, $\|x^*_i - z^*_i\|_{\infty} \leq |P|$ (cf. also Hochbaum, 1994, for a more general result).

To sum up, the above mentioned efficient procedures optimally partition storage space among SKUs if the forward area meets the quite typical and only mildly restricting precondition of equally sized shelves. In this case, it seems not meaningful to apply the fluid model as proposed by Hackman and Rosenblatt (1990) since it is neither much more efficiently solvable (though it requires only $O(|P|)$ time) nor leads to better solutions. Instead, as filling a single shelf with multiple products is often not manageable in the real-world, practitioners usually apply simple repairing techniques (mainly based on rounding) rather than the previously described method for ensuring integer solutions. Of course, such a repair bears the risk of non-optimal integer solutions as can be seen by the following small example, where three SKUs with aggregate restock costs $q_1 = 1$, $q_2 = 251$, and $q_3 = 1000$, respectively, have to be allocated to $S = 4$ shelves. Clearly, the optimal allocation is $x^* = (1, 1, 2)$ yielding $C_{\text{ESI}}(x^*) = 752$. In contrast, the fluid model solution according to Bitran and Hax’s algorithm is $z^* = (1, \sqrt[3]{251}, \sqrt[3]{1000}, \sqrt[3]{251} + \sqrt[3]{1000})$. Then, inappropriate repairing can lead to the solutions $(1, 2, 1)$ or $(2, 1, 1)$ and relative deviations in the objective function values of 49.8% and 66.4%, respectively, compared to the optimal value.

In the following section, we strengthen the previous argumentation by quantifying the resulting gap between the DFRAP_{ES} and its fluid counterpart FMB when applying different repair heuristics.

### 2.4 Computational study

#### 2.4.1 Repair heuristics

As existing research on the fluid model does not draw attention to repair heuristics which are typically applied by practitioners, we propose four different simple, intuitive, and easily applicable procedures for altering a non-integral solution gained with the fluid model to an integer solution. All repair heuristics are based on the following two steps:

1. Set $x^R_i = \lfloor z^*_i \rfloor$, i.e., round down non-integral solution elements of the fluid model.
2. In case $\sum_{i \in P} x^R_i < S$, adequately allocate all remaining free shelves to the products until all $S$ shelves are allocated. In case $\sum_{i \in P} x^R_i = S$, stop the procedure.

Clearly, after step [1] is performed, at least one shelf is allocated to each product and the total number of allocated shelves is integral. Thus, the solution is feasible for the
discrete problem, but, typically, the number of allocated shelves will be smaller than $S$. Note also that this solution is generally not optimal for the DFRAP\textsubscript{ES} with the right-hand side $S$ being replaced by $\sum_{i \in P} x^R_i$ (in contrast to the data given in Sect. 2.3). Let $\delta$ denote the difference between $S$ and the number of shelves allocated in Step 1, i.e., $\delta = S - \sum_{i \in P} x^R_i$. As $\sum_{i \in P} z^*_i = S$, it is readily verified that $\delta \in \{0, 1, \ldots, |P| - 1\}$. In order to yield a better estimate on the maximum number of iterations during Step 2, the co-domain of $\delta$ can be tightened to those values that are smaller than $|\{i \in P : z^*_i \not\in \mathbb{N}\}|$, i.e., the number of products to which the fluid model allocates non-integral numbers of shelves.

Clearly, in case $\delta > 0$ it is meaningful to allocate additional shelves to some SKUs as this improves the objection function value. This Step 2, i.e., the identification of promising SKUs, is specified next. Each of the four heuristical procedures will require exactly $\delta$ iterations and no SKU will receive more than one additional shelf per adaptation step:

**Procedure R1:** Randomly choose $\delta$ different SKUs according to a uniform distribution and increase the respective numbers of allocated shelves by one in each case.

**Procedure R2:** Rank all SKUs in order of non-decreasing $z^*_i$. According to this ranking increase the number of allocated shelves by one for each of the first $\delta$ SKUs.

**Procedure R3:** Rank all SKUs in order of non-increasing differences $z^*_i - x^R_i$. After that, increase the number of allocated shelves for each of the first $\delta$ SKUs by one.

**Procedure R4:** Rank all SKUs in order of non-increasing $d_i(x^R_i)$. Then, increase the number of allocated shelves for each of the first $\delta$ SKUs by one.

In case that a certain ranking is not unique, equal elements remain in their original order. For instance, assume $x^R = (1, 2, 1, 2, 1)$ after the first step. Then, the ranking of the SKUs according to $R2$ is $(1, 3, 5, 2, 4)$.

The repair heuristics are ordered in such a way, that they incorporate an increasing level of relevant information. $R1$ is a pure random procedure, $R2$ is implicitly based upon decreasing decrements $d_i(x_i)$ for all $i$ (considered isolated), and $R3$ constitutes standard rounding in integer programming. Finally, procedure $R4$ includes a (simpler) variant of the main idea underlying the incremental algorithm devised by Gross (1956) for the simple resource allocation problem and is assumed to perform best.

### 2.4.2 Setup of computational study

As there is no established test bed available, we now elaborate on the generation of test instances. As can be seen from the formulation of the DFRAP\textsubscript{ES}, the three relevant parameters are the number of products $|P|$, the total number of available shelves $S$, and the aggregate restock costs $q_i$ as introduced in Sect. 2.2.

As a forward area stores only fast moving SKUs, aggregate restock costs $q_i$, typically, do not vary substantially. Therefore, we scale $q_i$ to the interval $(0, 1]$ and assume three different product categories $P_1$, $P_2$, and $P_3$, and their corresponding probabilities $p_1, p_2$ and $p_3$, where $p_k$ defines the probability of a product belonging to category $P_k$ ($k = 1, 2, 3$). See Table 1 for the different distributions $D_1, D_2$, and $D_3$ of the product
categories. We assume the aggregate restock costs to be independent samples uniformly distributed in the intervals $[0.1, 0.2)$ (for $P_1$-products), $[0.2, 0.4)$ (for $P_2$-products), and $[0.4, 0.8)$ (for $P_3$-products).

<table>
<thead>
<tr>
<th>$D_1$</th>
<th>$P_1$</th>
<th>$P_2$</th>
<th>$P_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_2$</td>
<td>0.7</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td>$D_3$</td>
<td>0.3</td>
<td>0.4</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Table 1: Distributions of product categories

The forward area is assumed to consist of $S$ equally sized storage locations, e.g., lanes of gravity flow racks. We investigate differently sized forward areas, i.e., we choose $S \in \{50, 150, 500\}$. Finally, to analyze the impact of the number of products that are stored in the forward area, we assume a total number of $|P| = S/r$ products, where $r$ is varied as follows: $r \in \{5, 2, 1.25\}$. For each $(S, r)$-combination and distribution of the product categories according to Table 1 (leading to $3^3 = 27$ constellations) we generated 10,000 independent instances assuming $n_i = S - |P| + 1$ for each SKU $i$.

For each of these instances the discrete and the continuous optimum is determined as described above. Then, the four repair heuristics are applied to obtain an integer solution out of the fluid model solution. As performance indicators we evaluate the following data for each $(S, r, D)$-constellation:

- “AVG” denotes the relative gap between the objective function value of the discrete optimum $x^*$ and the respective “repaired” fluid model solution $x^R$ according to procedure $R \in \{R_1, R_2, R_3, R_4\}$, i.e., $(C_{ES1}(x^R) - C_{ES1}(x^*)) / C_{ES1}(x^*)$ averaged over all instances (in %),
- “MAX” defines the maximal relative deviation between the corresponding objective function values (in %),
- “DIF” denotes the relative number of instances for which $x^*$ and $x^R$ differ in at least one element, i.e., different number of shelves for at least one SKU (in %).

Note that for each generated instance the discrete solution is unique, which is due to the fact that the aggregate restock costs are i.i.d. drawn from a continuous probability distribution. Hence, DIF indicates the relative number of instances for which a certain repair heuristic generates a non-optimal solution, so that a DIF-value of 0 (100) means that the respective repaired solutions have always (never) been optimal for the discrete problem.

2.4.3 Results

Table 2 lists the results of our computational study on the DFRAP. Independently of the three distributions of the product categories, the procedures $R_4$ and $R_3$ perform
surprisingly well in terms of all measures. In particular, $R4$ always generates the optimal solution except for three ($D_1$-distributed) instances out of all 270,000 instances. Although the number of non-optimal solutions generated by $R3$ is higher than for $R4$, average relative deviations are negligible and maximal relative deviations are about 1%. If the product categories are not distributed according to $D_1$ and if $r = 1.25$, procedure $R3$ even always generates the optimal solution. Interestingly, in case $r = 2$, $R3$ performs worse than for the other two ratios.

<table>
<thead>
<tr>
<th>$S = 50$</th>
<th>$S = 150$</th>
<th>$S = 500$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r$</td>
<td>$RP$</td>
<td>AVG</td>
</tr>
<tr>
<td>5</td>
<td>$R1$</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td>$R2$</td>
<td>0.94</td>
</tr>
<tr>
<td></td>
<td>$R3$</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>$R4$</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>$R1$</td>
<td>6.25</td>
</tr>
<tr>
<td></td>
<td>$R2$</td>
<td>3.47</td>
</tr>
<tr>
<td></td>
<td>$R3$</td>
<td>0.34</td>
</tr>
<tr>
<td></td>
<td>$R4$</td>
<td>0.00</td>
</tr>
<tr>
<td>1.25</td>
<td>$R1$</td>
<td>8.68</td>
</tr>
<tr>
<td></td>
<td>$R2$</td>
<td>13.11</td>
</tr>
<tr>
<td></td>
<td>$R3$</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>$R4$</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 2: Results for DFRAP$_{ES}$

In contrast, the two other procedures $R1$ and $R2$ did not generate the optimal solution in almost any case. Both procedures are not competitive compared to $R4$ or $R3$. While average deviations of about 1% in case $r = 5$ are acceptable for $R1$ and $R2$, we observed maximal relative deviations up to 32% and average relative deviations up to 22% for $R2$. Obviously, the performance of $R1$ and $R2$ measured by AVG and MAX, respectively, deteriorates rather drastically as $r$ decreases for all tested distributions of
product categories. Interestingly, if \( r = 1.25 \), \( R2 \) performs considerably worse than the random procedure \( R1 \) in terms of AVG and MAX whereas \( R1 \) performs considerably worse than \( R2 \) in case \( r = 2 \) and slightly worse in case \( r = 5 \).

As procedure \( R4 \) performs outstandingly, we conducted some additional experiments testing further \((S, r, D)\)-constellations with key focus on maximal relative deviations between the objective function value of the \( R4 \)-solution and the optimal value. Actually, we did not observe instances of the \( \text{DFRAP}_{\text{ES}} \) that lead to relative deviations of more than 2.7%. Concerning procedure \( R3 \), we found instances for which the (maximal) relative deviation is just under 4% which is still passable.

From these experimental results we can deduce a quite important information for practitioners that stock their forward area using the fluid model. Although it is certainly better to apply one of the polynomial-time procedures stated in Sect. 2.2 and 2.3 that guarantee optimal solutions to \( \text{DFRAP}_{\text{ES}} \), a simple heuristic repair of fluid model solutions according to \( R4 \) will generally lead to optimal solutions for the discrete problem or at least to near-optimal ones for which the deviation from optimality can be ignored from a practical point of view.

3 The discrete forward-reserve assignment and allocation problem (DFRAAP)

In the previous DFRAP, the SKUs to be stored are predetermined and it only remains to decide in which quantities they are stored. In this section, we address the discrete forward-reserve assignment and allocation problem (DFRAAP). This problem simultaneously considers both the assignment of SKUs to the forward area and the allocation of storage space to each of the assigned SKUs.

3.1 DFRAAP with variable storage modes

The modeling of the DFRAAP is based on that of the DFRAP. Additionally, for each SKU \( i \) a storage mode \( j = 0 \) and its corresponding binary variable \( x_{i0} \) is introduced. This mode represents a SKU’s exclusion from the forward area, so that the respective variable indicates whether product \( i \) is assigned to the forward area \((x_{i0} = 0)\) or not \((x_{i0} = 1)\). If not assigned to the forward area, product \( i \) has to be picked from the reserve area, which results in an average cost of \( c_i^R \) per demand event. In contrast, a product \( i \) stocked in the forward area can more efficiently be picked at an average cost of \( c_F^i (< c_i^R) \), but has to be replenished from the reserve area, where each restock event is charged with cost \( c_i \). We assume that the total demand forecast \( d_i \) of a SKU \( i \) is spread on a given number of \( \alpha_i (\leq d_i) \) orders. The objective is to minimize the overall costs, i.e., for both picking and replenishing, per planning period. The model reads as:

\[
\text{DFRAAP: Minimize } C_2(x) = \sum_{i \in P} \sum_{j=1}^{n_i} \left( c_F^i \alpha_i + c_i \frac{d_i}{\alpha_{ij}} \right) x_{ij} + \sum_{i \in P} c_i^R \alpha_i x_{i0} \quad (21)
\]
subject to
\[ \sum_{j=0}^{n_i} x_{ij} = 1 \quad \forall i \in P \] (22)
\[ \sum_{i \in P} \sum_{j=1}^{n_i} w_{ij} x_{ij} \leq S \] (23)
\[ x_{ij} \in \{0, 1\} \quad \forall i \in P; j = 0, \ldots, n_i \] (24)

Again, this problem turns out to be \( \mathcal{NP} \)-hard in the ordinary sense as it is still mathematically equivalent to the multiple choice knapsack problem (MCKP, see Sect. 2.1). Nevertheless, the following structural property of DFRAAP-solutions can be applied for reducing the number of storage modes to be considered. Using equality \( \sum_{j=1}^{n_i} x_{ij} = 1 - x_{i0} \) (cf. (22)) for all \( i \), we rewrite the objective function (21) as follows:

\[ \sum_{i \in P} \sum_{j=1}^{n_i} \left( c_i^F o_i + c_i \frac{d_i}{a_{ij}} \right) x_{ij} + \sum_{i \in P} c_i^R x_{i0} = \sum_{i \in P} \sum_{j=1}^{n_i} c_i \frac{d_i}{a_{ij}} x_{ij} + \sum_{i \in P} (c_i^R - c_i^F) o_i x_{i0} + \sum_{i \in P} c_i^F o_i. \]

Let \( c_i^{add} = (c_i^R - c_i^F) \) denote the additional average pick cost incurred by picking SKU \( i \) from the reserve rather than the forward area. Then, minimizing \( C_3(x) \) is equivalent to minimizing

\[ \tilde{C}_2(x) = \sum_{i \in P} \sum_{j=1}^{n_i} c_i \frac{d_i}{a_{ij}} x_{ij} + \sum_{i \in P} c_i^{add} o_i x_{i0}. \] (25)

Obviously, objective function (25) reveals that it is not meaningful to store product \( i \) in forward mode \( j \) if the associated restock costs \( c_i d_i/a_{ij} \) are greater than the additional pick costs \( c_i^{add} o_i \), so that restock costs exceed the savings in picking. Let \( j_i^{min} > 0 \) denote the minimum (forward storage) mode, so that for the first time \( c_i d_i/a_{ij^{min}} < c_i^{add} o_i \) holds (if existing). Thus, no “smaller” storage mode, i.e., storing less than \( a_{ij^{min}} \) units of SKU \( i \) in forward area, than \( j_i^{min} \) needs to be considered in the DRFAAP. In case that \( j_i^{min} \) does not exist, we can directly conclude \( x_{i0} = 1 \).

3.2 DFRAAP with equally sized shelves

Analogously to the DFRAP, we now adapt the basic model of the previous section to the case with equally sized storage locations. By setting \( q_i^R = c_i^{add} o_i \), which denotes the total additional costs for picking all orders of SKU \( i \) from the reserve rather than the forward area, and recalling \( q_i = c_i d_i/a_i \) the DFRAAP\textsubscript{ES} can be formulated in its compact form as follows:

DFRAAP\textsubscript{ES}: Minimize \( C_{ES2}(x) = \sum_{i \in P} f_i(x_i) \) (26)
subject to

$$\sum_{i \in P} x_i \leq S$$  \hspace{1cm} (27)

$$x_i \in \{0, 1, \ldots, n_i\} \quad \forall i \in P;$$  \hspace{1cm} (28)

where

$$f_i(x_i) = \begin{cases} q_i/x_i & \text{if } x_i \geq 1, \\ q_i^R & \text{if } x_i = 0. \end{cases}$$  \hspace{1cm} (29)

Note that the objective function (26) is still separable but the $f_i$’s are not necessarily convex anymore. Clearly, the convexity of an $f_i$ depends on the minimum storage mode $j_{i}^{\min}$ of the respective SKU. A necessary but not sufficient condition for $f_i$ being convex is $j_{i}^{\min} = 1$. Only if

$$q_i^R - q_i \geq d_i(1)$$  \hspace{1cm} (30)

where $d_i(1)$ is defined by Eq. (12), then $f_i$ is convex. Thus, in case that all SKUs $i \in P$ satisfy condition (30), the DFRAAP$_{ES}$ turns out to be polynomial time solvable (cf. Sect. 2.2 and 2.3) as it is again mathematically equivalent to a simple discrete resource allocation problem with a separable convex objective function. In this case, the inequality-sign in Eq. (27) can be replaced by an equality-sign (if $\sum_{i \in P} n_i \geq S$).

Considering the more general case of not necessarily convex functions $f_i$, the following proofs deal with the complexity of solving the discrete assignment and allocation problem with equal shelves.

**Theorem 3.1**

The DFRAAP$_{ES}$ is ordinarily $\mathcal{NP}$-hard.

**Proof**

For proving the theorem we, first, have to show that the DFRAAP$_{ES}$ is $\mathcal{NP}$-hard. As the proof is based on a straightforward reduction from the well-known knapsack-problem, we only sketch the transformation scheme. The knapsack problem was shown to be $\mathcal{NP}$-hard by Karp (1972). For each knapsack item $i$ (and its associated weight $w_i > 0$ and profit $\delta_i > 0$) we introduce an SKU $i$, for which $w_i = n_i$, $q_i^R = M$ with $M > \delta_i w_i$, and $q_i = (M - \delta_i)w_i$ holds. Then, it is readily verified that $j_{i}^{\min} = n_i$ as $q_i^R - q_i/n_i = \delta_i$ and $q_i^R - q_i/j < 0$ for $j < n_i$. Thus, there exist only two meaningful storage modes for each SKU $i$ in this setting: allocating $n_i$ shelves or excluding the SKU by choosing 0 shelves, which corresponds to putting an item into the knapsack or not. Moreover, the savings realized when SKU $i$ is assigned to the forward area equal the profit of item $i$. Finally, we set the number of shelves $S$ available in the forward area to the knapsack capacity, which completes the transformation scheme.

Furthermore, the DFRAAP$_{ES}$ must be solvable in pseudo-polynomial time. Respective procedure are readily available in the literature. For instance, the pseudo-polynomial time dynamic programming procedure of Dudzinski and Walukiewicz (1987) for solving the MCKP, which is a generalization of our problem, can be applied. Alternatively, Ibaraki
and Katoh (1988, Sect. 3.3) present a dynamic programming procedure for the integer simple resource allocation problem with separable, non-convex objective function with $O(|P|S^2)$. However, for each given product selection (assignment) $A \subseteq P$ the resulting DFRAAPES can be solved efficiently (cf. Sect. 2.2 and 2.3). Thus, we can solve DFRAAPES to optimality by evaluating all possible SKU selections, i.e., at most $2^{|P|} - 1$ selections of SKUs, which results in solving an exponential number of DFRAAPES instances. Of course, this number can be reduced by enumerating only reasonable SKU assignments $A \subseteq P$, i.e., all $A$ fulfilling $\sum_{i \in A} j_i^{min} \leq S$. Further reductions might be yielded by some versatile branch-and-bound techniques. However, this is not within the scope of this paper but might be a topic for future research.

### 3.3 Setup of computational study

Analogously to the computational study described in Sect. 2.4, the following experiments compare the solutions of DFRAAPES and its (repaired) continuous counterparts in order to compute the gap between the fluid and the discrete model.

As suggested in the previous section, we solve an instance of the DFRAAPES and its respective fluid model by simply enumerating all reasonable SKU assignments $A \subseteq P$. Specifically, for each $A$ we determine:

- the discrete optimum using the algorithm of Bitran and Hax (1981) and the proximity between DFRAAPES and FMB as presented within Sect. 2.3 assuming $n_i = S - |A| + 1$ for all SKUs $i \in A$, and
- the fluid model solution according to Eq. (16) forcing $z_i \geq j_i^{min}$ for all $i \in A$ via Bitran and Hax’s algorithm. Clearly, the denominator in (16) is to be restricted to $A$.

After having determined both the discrete and the continuous optimal SKU assignment, we discretize the respective fluid model allocations according to the procedures $R2$ and $R4$, which turned out to be the worst and best performing non-random heuristical procedures in Sect. 2.4, respectively. This way, the gap between both problem versions can be computed.

To account for the exponential nature of our computational study, we investigate only two rather small values for $|P|$, i.e., we choose $|P| \in \{12, 24\}$. The number of shelves is set to $S = r|P|$, where $r$ is varied as follows: $r \in \{1/3, 1/2, 2/3\}$. Thus, it is guaranteed that not all SKUs can be assigned to the forward area. The assumptions on the aggregate restock costs $q_i$ are the same as in Sect. 2.4.2, i.e., we consider three product categories $P_1, P_2, P_3$, but only one distribution, namely $D_2$. Additionally, for the total additional costs we assume $q_i^R = \lambda_i q_i$, where $\lambda_i$ is uniformly distributed in the intervals $(0.1, 0.5)$ (with probability $p_\lambda$) and $(1.5, 2)$ (with the converse probability). We test the following three values $p_\lambda \in \{0.2, 0.5, 0.8\}$. Clearly, the coherence between $q_i$ and $q_i^R$ leads to $j_i^{min} > q_i/q_i^R = 1/\lambda_i$. 


For each \(|P|, r, p_\lambda\)-combination we generated 100 independent instances according to \(D_2\) (again, leading to \(3^3 = 27\) constellations) and recorded the relative number of instances (labeled as “SEL”) for which the optimal assignment for the continuous version differs from the optimal assignment for its discrete counterpart (in %). Note that SEL is independent of the repair heuristics. Additionally, we recorded the three evaluation measures AVG, MAX, and DIF (see Sect. 2.4) for each of the two repair heuristics.

### 3.4 Results

Table 3 contains the results of our computational study on the DFRAAP\(_{ES}\). Taking at first a look at the SEL-values, it becomes obvious that for given \(|P|\) and \(r\) the relative number of different optimal assignments between the discrete and the fluid model decreases as \(p_\lambda\) increases. For \(p_\lambda = 0.2\) we recorded up to 59% different assignments whereas SEL did not exceed 17% in case \(p_\lambda = 0.8\). Furthermore, except for three constellations, the SEL-values for \(|P| = 12\) are smaller than the respective values for \(|P| = 24\). Here, we suppose that the number of different assignments increases as the number of SKUs increases – particularly if \(p_\lambda\) is not too large.

| \(r\) | \(p_\lambda\) | \(|P| = 12\) | \(|P| = 24\) |
|------|---------|--------|--------|
|      | RP      | AVG    | MAX    | DIF    | SEL    | AVG    | MAX    | DIF    | SEL    |
| 1/3  | 0.2     | 3.04   | 7.87   | 86     | 46     | 2.96   | 6.67   | 100    | 45     |
|      | R4      | 0.34   | 2.62   | 46     |        | 0.14   | 1.33   | 45     |        |
|      | R2      | 4.35   | 11.92  | 87     | 26     | 4.87   | 9.22   | 99     | 38     |
|      | R4      | 0.25   | 3.07   | 26     |        | 0.19   | 1.18   | 38     |        |
|      | R2      | 1.48   | 10.89  | 32     | 7      | 2.83   | 10.56  | 65     | 17     |
|      | R4      | 0.10   | 5.80   | 7      |         | 0.05   | 0.81   | 17     |        |
| 1/2  | 0.2     | 5.35   | 11.78  | 99     | 27     | 6.10   | 11.05  | 100    | 50     |
|      | R4      | 0.14   | 2.47   | 27     |         | 0.12   | 0.66   | 50     |        |
|      | R2      | 4.84   | 17.24  | 85     | 17     | 7.07   | 17.21  | 99     | 45     |
|      | R4      | 0.12   | 2.91   | 17     |         | 0.16   | 1.39   | 45     |        |
|      | R2      | 2.09   | 21.88  | 45     | 9      | 1.39   | 10.25  | 83     | 8      |
|      | R4      | 0.07   | 2.25   | 9      |         | 0.02   | 0.59   | 8      |        |
| 2/3  | 0.2     | 7.59   | 14.88  | 97     | 38     | 8.76   | 14.28  | 100    | 59     |
|      | R4      | 0.15   | 1.90   | 38     |         | 0.19   | 1.16   | 59     |        |
|      | R2      | 5.27   | 15.57  | 74     | 25     | 6.58   | 18.62  | 99     | 44     |
|      | R4      | 0.14   | 3.09   | 25     |         | 0.13   | 1.03   | 44     |        |
|      | R2      | 1.07   | 11.67  | 47     | 4      | 0.72   | 3.95   | 83     | 4      |
|      | R4      | 0.01   | 0.47   | 4      |         | 0.01   | 0.51   | 4      |        |

Table 3: Results for DFRAAP\(_{ES}\)

Next, we evaluate the performance of the two repair heuristics \(R2\) and \(R4\) in case \(|P| = 12\). Obviously, average relative deviations (AVG) of \(R4\)-solutions as well as the number of different allocations (DIF) decrease as \(p_\lambda\) increases for a given \(r\). The same is true for \(R2\), except for one constellation. Interestingly, the development of the maximum relative deviations of \(R2\)-solutions is the other way round. Both procedures yield their maximum relative deviation of almost 22% (\(R2\)) and 6% (\(R4\)), respectively, in case \(p_\lambda = 0.8\). Regarding DIF-values again, also note that \(R4\) always generates the optimal solution whenever the optimal assignment for the fluid model is the same as for the discrete model (cf. DIF vs. SEL). In contrast, as expected, \(R2\) most likely generates a
non-optimal solution (cf. also Sect. 2.4.3) – especially if \( p_\lambda \) is small.

The results for \( |P| = 24 \) differ in some points from that obtained for \( |P| = 12 \). Now, smaller maximum relative deviations of almost 19% (\( R_2 \)) and 1.5% (\( R_4 \)), respectively, are yielded. However, on average, \( R_2 \) seems to perform poorer than before, whereas \( R_4 \) yields better AVG-values than in case \( |P| = 12 \), except for two constellations. Furthermore, \( R_2 \) generates non-optimal solutions in almost all tested cases where \( p_\lambda \neq 0.8 \). In contrast, again, \( R_4 \) generates optimal solutions whenever the fluid model and its discrete counterpart decide for the same assignment.

To sum up, if fluid model solutions are “repaired” according to \( R_4 \), acceptable deviations will generally be yielded. In contrast, the more intuitive and therefore probably more often applied heuristic \( R_2 \) performs considerably poorer in most cases. Like in Sect. 2.4.3 we conjecture that the loss in precision of the fluid model applied to the assignment and allocation problem will decrease if the number of available shelves is sufficiently large. In opposite to our managerial implications formulated there, it is now more advisable to apply the discrete model DFRAPES as an evaluation tool for comparing SKU selections instead of the fluid model irrespective of the repair heuristic as rather large deviations might follow in the latter case.

4 The discrete forward-reserve allocation and sizing problem (DFRASP)

In this section, we study another extension of the basic DFRAP by incorporating the size of the forward area as a decision variable. For a given set of products the area size has to be determined and partitioned among SKUs. Clearly, the smaller a forward area the less costly its (one-time) installation and daily operations. On the other hand, generally less SKUs can be stored forward, so that a more frequent replenishment is required.

4.1 DFRASP with variable storage modes

Based on the DFRAP defined in Sect. 2.1 let \( c_s \) denote the linear setup and operational costs incurred by any additional storage location, and let \( S_{\text{min}} \) and \( S_{\text{max}} \) denote the minimal and maximal size of the forward area, respectively. With these limits we implicitly account for the compact size of a forward area and thus, we assume negligible impacts on the forward area pick costs within these limits. Then, depending on \( c_s \) and the aggregate restock costs of each SKU, the DFRASP minimizes the overall costs incurred by replenishing and the area size while simultaneously deciding on the total number of shelves in the forward area and their optimal allocations to the given set of SKUs that have to be stored in the forward area. The respective model reads as follows:

\[
\text{DFRASP: Minimize } C_3(x, S) = \sum_{i \in P} \sum_{j=1}^{n_i} c_i \frac{d_{ij}}{a_{ij}} x_{ij} + c_s \cdot S
\]  

(31)
subject to
\[ \sum_{j=1}^{n_i} x_{ij} = 1 \quad \forall i \in P \]  
(32)

\[ \sum_{i \in P} \sum_{j=1}^{n_i} w_{ij} x_{ij} \leq S \]  
(33)

\[ x_{ij} \in \{0, 1\} \quad \forall i \in P; \ j = 1, \ldots, n_i \]  
(34)

\[ S \in \{S_{\text{min}}, \ldots, S_{\text{max}}\} \]  
(35)

Note that the problem is equivalent to the all-capacity multiple-choice knapsack problem (see Kellerer et al., 2004, Ch. 11), which – as a generalization of MCKP – is well-known to be \( \mathcal{NP} \)-hard. As the dynamic programming procedure of Dudzinski and Walukiewicz (1987) executed for \( S = S_{\text{max}} \) recursively determines an optimal knapsack for any smaller capacity \( S \leq S_{\text{max}} \), a pseudo-polynomial algorithm is readily available and the DFRASP turns out to be ordinarily \( \mathcal{NP} \)-hard.

4.2 DFRASP with equally sized shelves

When presupposing equally sized shelves the DFRASP\(_{\text{ES}}\) is defined as follows:

DFRASP\(_{\text{ES}}\): Minimize \[ C_{\text{ES3}}(x, S) = \sum_{i \in P} q_i x_i + c_s \cdot S \]  
(36)

subject to
\[ \sum_{i \in P} x_i = S \]  
(37)

\[ x_i \in \{1, \ldots, n_i\} \quad \forall i \in P \]  
(38)

\[ S \in \{S_{\text{min}}, \ldots, S_{\text{max}}\} \]  
(39)

Clearly, for any fixed size \( S \in \{S_{\text{min}}, \ldots, S_{\text{max}}\} \) where \( S_{\text{min}} \geq |P| \) and \( S_{\text{max}} \leq \sum_{i \in P} n_i \) the problem turns out to be equivalent to DFRASP\(_{\text{ES}}\), which is polynomial time solvable (cf. Sect. 2.2). Let \( x(S) = (x_1(S), \ldots, x_{|P|}(S)) \) denote the corresponding optimal allocations, we check in \( \mathcal{O}(|P|) \) time whether \( S \) is the optimal size or not. Therefore, we simply have to determine \( d_{\text{max}}(S) = \max\{d_i(x_i(S)) \mid i \in P, x_i(S) < n_i\} \) as well as \( d_{\text{min}}(S) = \min\{d_i(x_i(S)) - 1 \mid i \in P, x_i(S) > 1\} \). If \( d_{\text{max}}(S) > c_s \), the optimal size \( S^* \) is larger than the current size \( S \) and if \( d_{\text{min}}(S) < c_s \), we can conclude \( S^* < S \). In the other cases, where \( d_{\text{max}}(S) = c_s \) or \( d_{\text{min}}(S) = c_s \), an optimal solution to DFRASP\(_{\text{ES}}\) has been found. Thus, we can efficiently solve any DFRASP\(_{\text{ES}}\)-instance by application of binary search over the feasible forward area sizes which consumes at most \( \mathcal{O}(\log(S_{\text{max}} - S_{\text{min}})|P|^2 \log((S_{\text{max}} - |P|)/|P|)) \) time in total if at each iteration the corresponding DFRASP\(_{\text{ES}}\) is solved according to one of the fastest algorithms mentioned in Sect. 2.2. Clearly, in case \( S_{\text{min}} = |P| \) and \( S_{\text{max}} = \sum_{i \in P} n_i \) the effort of solving...
DFRASP\textsubscript{ES} reduces to $O(\sum_{i \in P} \log(n_i - 1))$ by efficiently determining for each SKU $i \in P$ the smallest decrement that is larger than $c_s$ (cf. also Frederickson and Johnson, 1982). Then, in case $n_i = n_j$ for all $i, j \in P$, note that

$$\sum_{i \in P} \log(n_i - 1) = \log \prod_{i \in P} (n_i - 1) = \log(n_i - 1)^{|P|} = |P| \log((S_{\text{max}} - |P|)/|P|)$$  \hspace{1cm} (40)

so that the asymptotic worst-case time complexity equals the one of an optimal algorithm for DFRAP\textsubscript{ES} (cf. Sect. 2.2).

### 4.3 Setup of computational study

Analogously to the experimental tests described in Sect. 2.4 and 3.3, the following experiments compare DFRASP\textsubscript{ES}-solutions and their (repaired) continuous counterparts in order to once again computing the gap between both models.

Problem DRFSAP\textsubscript{ES} is solved as proposed in the previous section assuming $n_i = S_{\max} - |P| + 1$. Its respective fluid model – where the relaxation concerns only the variables $x_i$ for all $i \in P$ – has been simply solved by applying the algorithm of Bitran and Hax (1981) for each feasible $S$ to determine the optimal fluid model size as well as the corresponding allocations. After having identified the continuous optimum, we heuristically discretize the respective allocations according to the procedures $R_2$ and $R_4$.

We investigated three different number of products, namely $|P| \in \{10, 25, 40\}$, and we set $S_{\min} = |P|$ and $S_{\max} = 3 \cdot |P|$. The assumptions on the aggregate restock costs $q_i$ for product $i$ are the same as in Sect. 2.4.2, i.e., we consider three product categories $P_1, P_2, P_3$, and, this time again, the three distributions $D_1, D_2, D_3$. Depending on the $q_i$’s we determined individual setup-costs according to $c_s = c \cdot \frac{(\sum_{i \in P}\sqrt{q_i})^2}{|P|^2}$ for each instance where we tested three different values for the parameter $c$, namely $c \in \{0.45, 0.75, 1.05\}$. The second multiplier is based on considerations for the classic fluid model (cf. Sect. 2.3) for which the optimal number of shelves

$$S_{FM}^* = \frac{\sum_{i \in P}\sqrt{q_i}}{c_s}$$  \hspace{1cm} (41)

(if no limits on $S$ are made a priori) is readily verified using Eq. (16) and differentiating the respective objective function $FC_{ES3}(z^*, S) = \sum_{i \in P} \frac{q_i}{z_i^*} + c_s \cdot S$ with respect to $S$. In this case, $c_s < \frac{(\sum_{i \in P}\sqrt{q_i})^2}{|P|^2}$ guarantees $S_{FM}^* > |P|$.

For each $(|P|, D, c)$-combination we generated 10,000 independent instances (again, leading to $3^3 = 27$ constellations) and recorded the relative number of instances (labeled as “SIZ”) for which the optimal size for the continuous version differs from the optimal size for its discrete counterpart (in %). Note that SIZ is independent of the repair heuristics. Additionally, we recorded the two evaluation measures AVG and MAX (see Sect. 2.4) for each of the two repair heuristics. Clearly, this time, AVG as well as MAX are not only related to the replenishing costs but to the total costs incurred by replenishing and setup.
4.4 Results

The results summarized in Table 4 reveal that the fluid model misses the right size of the forward area in a considerable fraction of instances. As indicated by performance measure SIZ, the fraction of properly sized areas deviates between 59.48% (for distribution $D_1$ and $c = 1.05$) and 0% (for distribution $D_3$ and $c = 1.05$).

| $|P| = 10$ | $|P| = 25$ | $|P| = 40$ |
|---|---|---|
| $c$ | RP | AVG | MAX | SIZ | AVG | MAX | SIZ | AVG | MAX | SIZ |
| $D_1$ | 0.45 | $R_2$ | 7.02 | 17.97 | 64.21 | 7.44 | 14.47 | 78.06 | 7.28 | 12.75 | 87.91 |
| | 0.75 | $R_4$ | 0.15 | 1.99 | 6.25 | 0.06 | 0.73 | 62.90 | 0.06 | 0.44 | 14.18 |
| | 1.05 | $R_4$ | 0.32 | 2.53 | 0.11 | 1.13 | 0.07 | 1.26 | 0.08 | 0.81 | 71.04 |
| $D_2$ | 0.45 | $R_2$ | 8.39 | 18.24 | 53.24 | 9.20 | 15.50 | 68.57 | 9.38 | 14.36 | 77.60 |
| | 0.75 | $R_4$ | 2.23 | 2.53 | 0.11 | 1.22 | 0.09 | 1.06 | 0.08 | 0.77 | 75.66 |
| | 1.05 | $R_4$ | 0.32 | 1.92 | 0.12 | 1.20 | 0.08 | 0.77 | 0.08 | 0.77 | 75.66 |
| $D_3$ | 0.45 | $R_2$ | 11.64 | 23.08 | 88.88 | 12.31 | 19.71 | 98.82 | 12.69 | 18.01 | 99.93 |
| | 0.75 | $R_4$ | 0.63 | 3.02 | 0.52 | 1.98 | 0.54 | 1.59 | 0.54 | 1.59 | 88.89 |
| | 1.05 | $R_4$ | 0.37 | 2.18 | 0.17 | 1.33 | 0.12 | 1.09 | 0.12 | 1.09 | 88.89 |

Table 4: Results for DFRASPES

When comparing the average and maximum gap between the DFRASP and its continuous counterpart (denoted as performance measures AVG and MAX in Table 4), rounding procedure $R_4$ again shows much more efficient than $R_2$. While $R_2$ produces a considerable average (maximum) gap ranging between 2.53% (5.11%) and 12.69% (23.08%), $R_4$ leads to an average gap of less then 1% in any parameter constellation and a maximum gap of only 3.02%. To conclude, when utilizing an appropriate repair procedure the practitioner applying the fluid model receives near optimal solutions; whereas the resulting size of the forward area will most probably deviate from the optimal area size.

5 Conclusions

Unlike existing research, which predominantly focuses the fluid model, the paper on hand investigates three discrete forward-reserve problems. Problem DFRAP allocates the storage space among a given set of SKUs and problem DFRAAP additionally selects the products to be stored in the forward area. Within the DFRASP jointly with the allocation problem it is decided on the overall size of the forward area. All resulting problem versions are investigated for two different constellations, namely variable storage modes and equally sized shelves. While the former case allows each SKU being stored in facultative modes each requiring a specific amount of space for a specific number of products, the latter scenario presupposes some standardized rack being subdivided into
equally sized shelves. The complexity results for all six resulting problem versions are summarized in Table 5. Considering equally sized shelves, in three separate computational studies we, furthermore, quantify the resulting gap between the respective discrete problem version and its continuous counterpart repaired to an integral solution.

<table>
<thead>
<tr>
<th>variable storage modes</th>
<th>equally sized shelves</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFRAP</td>
<td>(\mathcal{O}(\max{</td>
</tr>
<tr>
<td>DFRAAP</td>
<td>ordinarily (\mathcal{NP})-hard (MCKP)</td>
</tr>
<tr>
<td>DFRASP</td>
<td>ordinarily (\mathcal{NP})-hard (all-capacity MCKP)</td>
</tr>
<tr>
<td></td>
<td>(\mathcal{O}(\log(S_{\text{max}} - S_{\text{min}})</td>
</tr>
</tbody>
</table>

Table 5: Summary of results

Future research should further concentrate on the discrete versions of the forward-reserve problem. For instance, a powerful branch-and-bound algorithm for the DFRAAP based on the ideas of this paper would be a valuable contribution. Furthermore, relevant problem extensions should be investigated. For example, explicitly considering the impact of the location of an SKU in the forward area on the resulting picking effort and explicitly scheduling replenishment activities bear plenty opportunities for future research.

Acknowledgments

The authors are grateful to Malte Fliedner for some hints on the resource allocation problem.

References


21


