Don't Rock the Boat: Algorithms for Balanced Dynamic Loading and Unloading

Sándor P. Fekete[1](✉), Sven von Höveling[1], Joseph S. B. Mitchell[2], Christian Rieck[1], Christian Scheffer[1], Arne Schmidt[1], and James R. Zuber[2]

[1] Department of Computer Science, TU Braunschweig, 38106 Braunschweig, Germany
{s.fekete,v.sven,c.rieck,c.scheffer,arne.schmidt}@tu-bs.de
[2] Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, NY 11794, USA
joseph.mitchell@stonybrook.edu, zuber139@gmail.com

Abstract. We consider dynamic loading and unloading problems for heavy geometric objects. The challenge is to maintain balanced configurations at all times: minimize the maximal motion of the overall center of gravity. While this problem has been studied from an algorithmic point of view, previous work only focuses on balancing the *final* center of gravity; we give a variety of results for computing balanced loading and unloading schemes that minimize the maximal motion of the center of gravity during the entire process.

In particular, we consider the one-dimensional case and distinguish between *loading* and *unloading*. In the unloading variant, the positions of the intervals are given, and we search for an optimal unloading order of the intervals. We prove that the unloading variant is NP-complete and give a 2.7-approximation algorithm. In the loading variant, we have to compute both the positions of the intervals and their loading order. We give optimal approaches for several variants that model different loading scenarios that may arise, e.g., in the loading of a transport ship with containers.

1 Introduction

Packing a set of objects is a classic challenge that has been studied extensively, from a variety of perspectives. The basic question is: how can the objects be arranged to fit into a container? Packing problems are important for a large spectrum of practical applications, such as loading items into a storage space, or containers onto a ship. They are also closely related to problems of scheduling and sequencing, in which issues of limited space are amplified by including temporal considerations.

Due to space constraints, several technical details are omitted from this extended abstract. A full version with all proofs can be found at [7].

M. A. Bender et al. (Eds.): LATIN 2018, LNCS 10807, pp. 448–460, 2018.
https://doi.org/10.1007/978-3-319-77404-6_33

Packing and scheduling are closely intertwined in *loading* and *unloading* problems, where the challenge is not just to compute an acceptable *final* configuration, but also the process of *dynamically building* this configuration, such that intermediate states are both achievable and stable. This is highly relavant in the scenario of loading and unloading container ships, for which maintaining *balance* throughout the process is crucial (Fig. 1).

Balancedness of packing also plays an important role for other forms of shipping: Mongeau and Bes [14] showed that displacing the center of gravity by less than 75 cm in a long-range aircraft may cause, over a 10,000 km flight, an additional consumption of 4,000 kg of fuel.

Fig. 1. Loading and unloading container ships.

In this paper, we consider algorithmic problems of balanced loading and unloading. For unloading, this means planning an optimal sequence for removing a given set of objects, one at a time; for loading, this requires planning both position and order of the objects.

The practical constraints of loading and unloading motivate a spectrum of relevant scenarios. As ships are symmetric around their main axis, we focus on one-dimensional settings, in which the objects correspond to intervals. Containers may be of uniform size, but stackable up to a certain limited height; because sliding objects on a moving ship are major safety hazards, stability considerations may prohibit gaps between containers. On the other hand, containers of extremely different size pose particularly problematic scenarios, which is why we also provide results for sets of containers whose sizes are exponentially growing.

1.1 Our Contributions

Our results are as follows; throughout the paper, *items* are the objects that need to be loaded (also sometimes called *placed*) or unloaded, while *container* refers to the space that accomodates them. Furthermore, we assume all objects to have unit density, i.e., their weights correspond to their lengths. In most cases, items correspond to geometric intervals.

- For unloading, we show that it is NP-complete to compute an optimal sequence. More formally, given a set of placed intervals $\{I_1, \dots, I_n\}$, it is NP-complete to compute an order $\langle I_{\pi(1)}, \dots, I_{\pi(n)} \rangle$, in which intervals are removed one at a time, such that the maximal deviation of the gravity's center is minimized.
- We provide a corresponding polynomial-time 2.7-approximation algorithm. In particular, we give an algorithm that computes an order of the input intervals such that removing the intervals in that order results in a maximal deviation which is no larger than 2.7 times the maximal deviation induced by an optimal order.
- For loading, we give a polynomial-time algorithm for the setting in which gaps are not allowed. In particular, given a set of lengths values $\ell_1, \dots, \ell_n \in \mathbb{R}_{>0}$, we require a sequence $\langle I_{\pi(1)}, \dots, I_{\pi(n)} \rangle$ of pairwise disjoint intervals with $|I_{\pi(i)}| = \ell_{\pi(i)}$ for $i = 1, \dots, n$ such that the following holds: Placing the interval $I_{\pi(i)}$ in the i-th step results in an n-stepped loading process such that the union of the loaded intervals is connected for all points in time during the loading process. Among these *connected placements*, we compute one for which the maximal deviation of the center of gravity is minimized.
- We give a polynomial-time algorithm for the case of stackable unit intervals. More formally, given an input integer $\mu \geq 1$, in the context of the previous variant, we relax the requirement that the union of the placed intervals has to be connected and additionally allow that the placed intervals may be stacked up to a height of μ in a stable manner, defined as follows. We say that layer 0 is completely *covered.* An interval I can be placed, i.e., covered, in layer $k \geq 1$ if the interval I is covered in all layers $0, \dots, k-1$ and if I does not overlap with another interval already placed in layer k.
- We give a polynomial-time algorithm for the case of exponentially growing lengths. More formally, in the context of the previous variant, we require that all intervals are placed in layer 1 and assume that the lengths of the input

intervals' lengths are exponentially increasing, i.e., there is an $x \geq 2$ such that $x \cdot \ell_i = \ell_{i+1}$ holds for all $i \in \{1, \dots, n-1\}$.

1.2 Related Work

Previous work on cargo loading covers a wide range of specific aspects, constraints and objectives. The general Cargo Loading Problem (CLP) asks for an optimal packing of (possibly heterogeneous) rectangular boxes into a given bin, equivalent to the Cutting Stock Problem [10]. Most of the proposed methods are heuristics based on (mixed) integer programming and have been studied both for heterogeneous or homogeneous items. Bischoff and Marriott [2] show that the performances of some heuristics may depend on the kind of cargo.

Amiouny et al. [1] consider the problem of packing a set of one-dimensional boxes of different weight and different length into a flat bin (so they are not allowed to stack these boxes), in such a way that after placing the last box, the center of gravity is as close as possible to a fixed target point. They prove strong NP-completeness by a reduction from 3-Partition and give a heuristic with a guaranteed accuracy within $\ell_{max}/2$ of a given target point, where ℓ_{max} is the largest box, w.r.t. length. A similar heuristic is given by Mathur [13].

Gehring et al. [9] consider the general CLP, for which (rectangular) items may be stacked, and place them in any possible position. They construct non-intersecting *walls*, i.e. packings made from similar items for slices of the original container, to generate the overall packing. They also show that this achieves a good final balancing of the loaded items. Mongeau and Bes [14] consider a similar variant for which the objective is to maximize the loaded weight. In addition, there may be other parameters, e.g., each item may have a different priority [22]. A mixed integer programming approach on this variant is given by Vancroonenburg et al. [23]. Limbourg et al. [11] consider the CLP based on the moment of inertia. Gehring and Bortfeldt [8] give a genetic algorithm for *stable* packings. Fasano [6] considers packing problems of three-dimensional *tetris*-like items in combination with balancing constraints. His work is done within the context of the Automated Transfer Vehicle, which was the European Space Agency's transportation system supporting the International Space Station (ISS).

Another variant is to consider multiple *drops*, for which loaded items have to be available at each drop-off point in such a way that a rearrangement of the other items is not required; see e.g. [3,4,12]. Davies and Bischoff [5] propose an approach that produces a high space utilization for even weight distribution. These scenarios often occur in container loading for trucks, for which the objective is to achieve an even weight distribution between the axles. For a state-of-the-art survey of vehicle routing with different loading constraints and a spectrum of scenarios, see Pollaris et al. [19].

In the context of distributing cargo by sea, two different kind of ships are distinguished: *Ro-Ro* and *Lo-Lo* ships. Ro-Ro (for roll on–roll off) ships carry wheeled cargo, such as cars and trucks, which are driven on and off the ship. Some approaches and problem variants such as multiple drops, additional loading, and optional cargo as well as routing and scheduling considering Ro-Ro ships are considered by Øvstebø et al. [15,16]. On the other hand, Lo-Lo (load on–load off) ships are cargo ships that are loaded and unloaded by cranes, so any feasible position can be directly reached from above.

While all this work is related to our problem, it differs by not requiring the center of gravity to be under control for each step of the loading or unloading process. A problem in which such a constraint is required and permanently checked is Compact Vector Summation (CVS), which asks for permutation to sum a number of k-dimensional vectors in a way that keeps each partial sum within a bounded k-dimensional ball. See Sevastianov [20,21] for a summary of results in CVS and its application in job scheduling. A different (and somewhat less serious) angle is considered by Paterson and Zwick [18] and Paterson et al. [17], who consider maximizing the reach beyond the edge of a table by stacking n identical, homogeneous, frictionless bricks of length 1 without toppling over, corresponding to keeping the center of gravity of subarrangements supported.

2 Preliminaries

An *item* is a unit interval $I := [m-\frac{1}{2}, m+\frac{1}{2}]$ with midpoint m. A set $\{I_1, \ldots, I_n\}$ of n items with midpoints $m_1, \ldots, m_n$ is *valid* if $m_i = m_j$ or $|m_i - m_j| \geq 1$ holds for all $i, j = 1, \ldots, n$. The *center of gravity* $C\left(I_1, \ldots, I_n\right)$ of a valid set $\{I_1, \ldots, I_n\}$ of items is defined as $\frac{1}{n}\sum_{i=1}^{n} m_i$.

For given a valid set $\{I_1, \ldots, I_n\}$ of items we seek orderings in which each item I_j is removed or placed such that the maximal deviation for all points in time $j = 1, \ldots, n$ is minimized. More formally, for $j = 1, \ldots, n$ and a permutation $\pi : j \mapsto \pi_j$, let $C_j := C\left(I_{\pi_j}, \ldots, I_{\pi_n}\right)$.

The Unloading Problem (Unload) seeks to minimize the maximal deviation during an unloading process of $I_1, \ldots, I_n$. In particular, given an input set $\{I_1, \ldots, I_n\}$ of items, we seek a permutation π such that $\max_{i,j=1,\ldots,n} |C_i - C_j|$ is minimized.

In the Loading Problem (Load) we relax the constraint that the positions of the considered items are part of the input. In particular, we seek an ordering and a set of midpoints for the containers such that the containers are disjoint and the maximal deviation for all points in time of the loading process is minimized; see Sect. 4 for a formal definition.

3 Unloading

We show that the problem Unload is NP-complete and give a polynomial-time 2.7-approximation algorithm for Unload. We first show that there is a polynomial- time reduction from the discrete version of Unload, the Discrete Unloading Problem (dUnload), to Unload; this leads to a proof that Unload is NP-complete, followed by a 2.7-approximation algorithm for Unload.

In the Discrete Unloading Problem (dUnload), we do not consider a set of items, i.e., unit intervals, but a discrete set $X := \{x_1, \ldots, x_n\}$ of points.

The center of gravity $C(X)$ of X is defined as $\frac{1}{n}\sum_{i=1}^{n} x_i$. For $j = 1, \ldots, n$ and a permutation $\pi : j \mapsto \pi_j$, let $C_j = C\left(x_{\pi_j}, \ldots, x_{\pi_n}\right)$. Again, we seek a permutation such that $\max_{i,j=1,\ldots,n} |C_i - C_j|$ is minimized.

Corollary 1. UNLOAD *and* dUNLOAD *are polynomial-time equivalent.*

3.1 NP-Completeness of the Discrete Case

We can establish NP-completeness of the discrete problem dUNLOAD.

Theorem 1. dUNLOAD *is* NP*-complete.*

The proof is based on a reduction of 3-PARTITION and omitted for lack of space; see the full version of this paper [7]. Because of the polynomial-time equivalance of dUNLOAD and UNLOAD, we conclude the following.

Corollary 2. UNLOAD *is* NP*-complete.*

3.2 Lower Bounds and an Approximation Algorithm

When unloading a set of items, their positions are fixed, so (after reversing time) unloading is equivalent to a loading problem with predetermined positions. For easier and uniform notation throughout the paper, we use this latter description.

In order to develop and prove an approximation algorithm for dUNLOAD, we begin by examining lower bounds on the span, $R - L$, of a minimal interval, $[L, R]$, containing the centers of gravity at all stages in an optimal solution.

Without loss of generality, we assume that the input points x_i sum to 0 (i.e., $\sum_i x_i = 0$), so that the center of gravity, C_n, of all n input points is at the origin. We let $R = \max_i C_i$ and $L = \min_i C_i$. Our first simple lemma leads to a first (fairly weak) bound on the span.

Lemma 1. *Let $(x_1, x_2, x_3, \ldots)$ be any sequence of real numbers, with $\sum_i x_i = 0$. Let $C_j = (\sum_{i=1}^{j} x_i)/j$ be the center of gravity of the first j numbers, and let $R = \max_i C_i$ and $L = \min_i C_i$. Then, $|R - L| \geq \frac{|x_i|}{i}$, for all $i = 1, 2, \ldots$.*

Due to space constraints, the proof of Lemma 1 is omitted; it can be found in the full version of this paper [7].

Corollary 3. *For any valid solution to* dUNLOAD*, the minimal interval $[L, R]$ containing the center of gravity at every stage must have length $|R - L| \geq \frac{|u_i|}{i}$ where u_i is the input point with the i-th smallest magnitude.*

We note that the naive lower bound given by Corollary 3 can be far from tight: Consider the sequence $1, 2, 3, 4, 5, 6, 7, -7, -7, -7, -7$. In the optimal order, the first -7 is placed fourth, after $2, 1, 3$. The optimal third and fourth centers, $\{2, -\frac{1}{4}\}$ are the largest magnitude positive and negative centers seen, and show a span 2.25 times greater than the naive bound of 1. By placing the first -7

in the third position, $R \geq \frac{3}{2}$, and $L \leq -\frac{4}{3}$. By placing it fifth, $R \geq \frac{5}{2}$. Our observation was that failing to place our first -7 if the cumulative sum is > 7 would needlessly increase the span.

This generalizes to the sequence $(x_1 = 1, x_2 = 2, \ldots, x_{k-1} = k-1, x_k = -k, x_{k+1} = -(k+1), \ldots, x_N)$, with an appropriate x_N to make $\sum x_i = 0$. If we place positive weights in increasing order until $c_l \geq \frac{k}{l}$, placing $-k$ instead of a positive at position l would decrease the center of gravity well below $\frac{k}{l}$. The first negative should be placed when $\min_l \frac{l^2-l}{2} \geq k$, which is when $l \approx \sqrt{2k}$. In this example, our optimal center of gravity span is at least $\frac{k}{l} \approx \sqrt{\frac{k}{2}}$, not the 1 from the naive bound of Corollary 3.

We now describe our heuristic, $\mathcal{H}$, which leads to a provable approximation algorithm. It is convenient to relabel and reindex the input points as follows. Let $(P_1, P_2, \ldots)$ denote the positive input points, ordered (and indexed) by increasing value. Similarly, let $(N_1, N_2, \ldots)$ denote the negative input points, orders (and indexed) by increasing magnitude $|N_i|$ (i.e., ordered by decreasing value).

The heuristic $\mathcal{H}$ orders the input points as follows. The first point is simply the one closest to the origin (i.e., of smallest absolute value). Then, at each step of the algorithm, we select the next point in the order by examining three numbers: the partial sum, S, of all points placed in the sequence so far, the smallest magnitude point, α, not yet placed that has the same sign as S, and the smallest magnitude point, β, not yet placed that has the opposite sign of S. If $S + \alpha + \beta$ is of the same sign as S, then we place β next in the sequence; otherwise, if $S + \alpha + \beta$ has the opposite sign as S, then we place α next in the sequence. The intuition is that we seek to avoid the partial sum from drifting in one direction; we switch to the opposite sign sequence of input points in order to control the drift, when it becomes expedient to do so, measured by comparing the sign of S with the sign of $S+\alpha+\beta$, where α and β are the smallest magnitude points available in each of the two directions. We call the resulting ordering the $\mathcal{H}$-permutation. The $\mathcal{H}$-permutation puts the j-th largest positive point, P_j, in position π_j^+ in the order, and puts the j-th largest in magnitude negative point, N_j, in position π_j^- in the order, where

$$\pi_j^+ = j + \max_k \{k \ : \ \sum_{i=1}^{k} |N_i| \leq \sum_{l=1}^{j} P_l\} \text{ and } \pi_j^- = j + \max_k \{k \ : \ \sum_{i=1}^{k} P_i < \sum_{l=1}^{j} |N_l|\}.$$

We obtain an improved lower bound based on our heuristic, $\mathcal{H}$, which orders the input points according to the $\mathcal{H}$-permutation.

Lemma 2. *A lower bound on the optimal span of* dUnload *is given by* $|R - L| \geq \frac{P_i}{\pi_i^+}$ *and* $|R - L| \geq \frac{|N_i|}{\pi_i^-}$.

To prove the lemma, we begin with a claim.

Claim. For any input set to the discrete unloading problem, where s_i are all terms with the same sign sorted by magnitude, a permutation π that minimizes the maximum value of the ratio $\frac{|s_i|}{\pi_i}$ must satisfy $\pi_k < \pi_i$, for all $k < i$.

Proof. By contradiction, assume that the minimizing permutation π has the maximum value of the ratio $\frac{|s_i|}{\pi_i}$ occur at an i for which there exists a $k < i$ for which $\pi_i \leq \pi_k$, which means that $\pi_i < \pi_k$ (because π_i cannot equal π_k for a permutation π, and $k \neq i$).

Because the terms s_i are indexed in order sorted by magnitude, $|s_k| \leq |s_i|$. Exchanging the order of s_i and s_k in the permutation would lead to two new ratios in our sequence: $\frac{|s_i|}{\pi_k}$ and $\frac{|s_k|}{\pi_i}$. Because $\pi_k > \pi_i$, we get $\frac{|s_i|}{\pi_k} < \frac{|s_i|}{\pi_i}$. Because $|s_k| \leq |s_i|$, we get $\frac{|s_k|}{\pi_i} \leq \frac{|s_i|}{\pi_i}$. Because these new ratios are smaller than $\frac{|s_i|}{\pi_i}$, we get a contradiction to the fact that π minimizes the maximum ratio.

The following claim is an immediate consequence of Lemma 1.

Claim. For the i maximizing $\frac{P_i}{\pi_i^+}$, any ordering placing this element earlier than π_i^+ in the sequence has a span $|R - L| > \frac{P_i}{\pi_i^+}$. Similarly, for the i maximizing $\frac{|N_i|}{\pi_i^+}$, any ordering placing this element earlier than π_i^- in the sequence has a span $|R - L| > \frac{|N_i|}{\pi_i^-}$.

On the other hand, the following holds.

Claim. For the i maximizing $\frac{P_i}{\pi_i^+}$, any ordering placing this element later than π_i^+ in the sequence has a span $|R - L| > \frac{P_i}{\pi_i^+}$. A similar statement holds for $\frac{|N_i|}{\pi_i^-}$.

Proof. The proof is by contradiction. The index into the $\mathcal{H}$ permutation maximizing the ratio $\frac{|x_k|}{k}$ is i. We assume (wlog) $x_i = P_J > 0$, and we let $K = i - J$.

If P_J is not placed in position i, we suppose another element, x, can be placed in its stead and results in a span that is less than $\frac{P_J}{i}$.

When placing any positive $x > P_J$ in the initial i position, the lowest possible observed span from Lemma 1 is at least $\frac{x}{i} > \frac{P_J}{i}$, which would contradict our assumption. Similarly, all positive points placed before or at position i must be less than or equal to P_J.

All permutations of these $J - 1$ positive elements and the first $K + 1$ negative elements have a large negative center of gravity at position i. From $K = \max_k\{k : \sum_{i=1}^{k} |N_i| \leq \sum_{l=1}^{J} P_l\}$, we get $\sum_{i=1}^{K+1} |N_i| \geq \sum_{l=1}^{J} P_l$, and hence $\sum_{i=1}^{K+1} N_i + \sum_{l=1}^{J} P_l \leq 0$, implying $\sum_{i=1}^{K+1} N_i + \sum_{l=1}^{J-1} P_l \leq -P_J$. Therefore, the maximizing value satisfies

$$|c^*| = \frac{|\sum_{i=1}^{K+1} N_i + \sum_{l=1}^{J-1} P_l|}{i} \geq \frac{P_J}{i}$$

Because the center of gravity is at a location greater than the $\mathcal{H}$-bound, and $R \geq 0 \geq L$, this span is also greater than the $\mathcal{H}$-bound and we can neither place an element greater than P_J nor one less than P_J in place of P_J while lowering the span beneath the $\mathcal{H}$-bound.

Theorem 2. *The $\mathcal{H}$-permutation minimizes the maximum (over i) value of the ratio $\frac{|x_i|}{\pi_i}$, and thus yields a lower bound on $|R - L|$.*

For the worst-case ratio, we get the following.

Theorem 3. *The $\mathcal{H}$ heuristic yields an ordering having span $R - L$ at most 2.7 times larger than the $\mathcal{H}$-lower bound.*

Due to space constraints, the proof of Theorem 3 is omitted; it can be found in the full version of this paper [7].

Corollary 4. *There is a polynomial-time* 2.7*-approximation algorithm for* Unload*.*

4 Loading

We proceed to loading problems, which requires a wider range of definitions: The positions of the objects are part of the optimization and for some loading variants, the items may have different lengths. Consider the following more general definitions:

An *item* is given by a real number ℓ. By assigning a *position* $m \in \mathbb{R}$ to an item, we obtain an interval I with length ℓ and midpoint m. For $n \geq 1$, we consider a set $\{\ell_1, \ldots, \ell_n\}$ of n items and assume $\ell_1 \geq \cdots \geq \ell_n$. Furthermore, $\{\ell_1, \ldots, \ell_n\}$ is *uniform* if $\ell := \ell_1 = \ldots = \ell_n$.

A *state* is a set $\{(I_1, h_1), \ldots, (I_n, h_n)\}$ of pairs, each one consisting of an interval I_j and an integer $h_j \geq 1$, the *layer* in which I_j lies. A state satisfies the following: (1) Two different intervals that lie in the same layer do not overlap and (2) for $j = 2, \ldots, n$, an interval in layer j is a subset of the union of the intervals in layer $j-1$.

A state $\{(I_1, h_1), \ldots, (I_n, h_n)\}$ is *plane* if all intervals lie in the first layer.

To simplify the following notations, we denote for $j = \{1, \ldots, n\}$ the midpoint of the interval I_j by m_j. The *center of gravity* $C(s)$ of a state $s = \{(I_1, h_1), \ldots, (I_n, h_n)\}$ is defined as $\frac{1}{M}\sum_{j=1}^{n} \ell_j m_j$, where M is defined as $\sum_{j=1}^{n} \ell_j$.

A *placement* p of an n-system S is a sequence $\langle I_1, \ldots, I_n \rangle$ such that $\{(I_1, h_1), \ldots, (I_j, h_j)\}$ is a state, the *j-th state* s_j, for each $j = 1, \ldots, n$. The 0-th state s_0 is defined as $\varnothing$ and its center of gravity $C(s_0)$ is defined as 0.

Definition 1. *The* loading problem *(*Load*) is defined as follows: Given a set of n items, we are searching for a placement p such that the $n+1$ centers of gravity of the $n+1$ states of p lie close to* 0*. In particular, the* deviation *$\Delta(p)$ of a placement p is defined as* $\max_{j=0,\ldots,n} |C(s_j)|$*. We seek a placement of S with minimal deviation among all possible placements for S.*

We say that stacking is not allowed *if we require that all intervals are placed in layer* 1*. Otherwise, we say that* stacking is allowed*. For a given integer $\mu \geq 1$ we say that μ is the* maximal stackable height *if we require that all used layers are no larger than μ.*

Note that in the loading case, minimizing the deviation is equivalent to minimizing the diameter, i.e., minimizing the maximal distance between the smallest and largest extent of the centers.

4.1 Optimally Loading of Unit Items with Stacking

Now we consider the case of unit items for which stacking is allowed. We give an algorithm that optimally loads a set of unit items with stacking.

Theorem 4. *There is a polynomial-time algorithm for loading a set of unit items so that the deviation of the center of gravity is in* $[0, \frac{1}{1+\mu}]$*, where* μ *is the maximum stackable height.*

Proof. Let m_i be the midpoint of item ℓ_i. Because we are allowed to stack items up to height μ, the strategy is the following: set $m_1 = m_2 = \cdots = m_\mu = \frac{1}{1+\mu}$, i.e., the first μ items are placed at the very same position. Call these first μ items the *starting stack* $\mathcal{S}_0$. Subsequently, we place the following items on alternating sides of $\mathcal{S}_0$, i.e., the item $\ell_{\mu+1}$ is placed as close as possible on the left side of $\mathcal{S}_0$, $\ell_{\mu+2}$ is placed as close as possible on the right side, $\ell_{\mu+3}$ is placed on top of $\ell_{\mu+1}$ (if we did not already reach the maximum stackable height of μ), or next to $\ell_{\mu+1}$ (if $\ell_{\mu+1}$ is on the μ-th layer), etc.

After each placement of $\ell_i, 1 \leq i \leq \mu$, we have $C(\ell_i) = \frac{1}{1+\mu}$. After two more placed items, the center of gravity is again at $\frac{1}{1+\mu}$, because these items neutralize each other. Thus, the critical part is a placement on the left side of $\mathcal{S}_0$. We proceed to show that after placing an item on the left side, the center of gravity is at position at least 0.

The midpoint $m_{\mu+1}$ of the item $\ell_{\mu+1}$ is $\frac{-\mu}{1+\mu}$, thus $C(\ell_{\mu+1}) = \frac{\mu}{1+\mu} - \frac{\mu}{1+\mu} = 0$. Now assume that we have already placed $c = (2k+1)\cdot\mu+\zeta$ items, where $\zeta < 2\mu$ and odd, i.e., we have already placed the starting stack $\mathcal{S}_0$ and k additional stacks of height μ on each side of $\mathcal{S}_0$. Let $z := (2k+1)\cdot\mu$. Then the center of gravity is at position $C(c)$, where

$$\begin{aligned} C(c) =& \frac{z\cdot\frac{1}{1+\mu} + \sum_{i=z+1}^{z+\zeta} m_i}{z+\zeta} = \frac{(z+\zeta-1)\cdot\frac{1}{1+\mu} + \frac{-k\mu-k-\mu}{1+\mu}}{z+\zeta} = \frac{k\mu+\zeta-1-k}{(1+\mu)\cdot(z+\zeta)} \\ =& \frac{k(\mu-1)+\zeta-1}{(1+\mu)\cdot(z+\zeta)} \geq \frac{\zeta-1}{(1+\mu)\cdot(z+\zeta)} \geq \frac{0}{(1+\mu)\cdot(z+\zeta)} \geq 0. \end{aligned}$$

In the following we show that there is no strategy that can guarantee a smaller deviation of the center of gravity than the strategy described in the last theorem.

Theorem 5. *The strategy given in Theorem 4 is optimal for* $n > \mu$*, i.e., there is no strategy such that the center of gravity deviates in* $[0, \frac{1}{1+\mu})$*.*

Proof. Because $n > \mu$, we must use at least two stacks. Now assume that we first place k items on one stack $\mathcal{S}_0$, before we start another one. Without loss of generality, we place this first k items at position $\frac{1}{1+\mu} - \varepsilon$. We proceed to show that for any $\varepsilon > 0$, we need k to be at least $\mu+1$, to get the new center of gravity to position $> -\varepsilon$ and therefore a smaller deviation as the strategy in Theorem 4.

If we place the item ℓ_{k+1} on the right side of $\mathcal{S}_0$, the new center of gravity gets to a position larger than $\frac{1}{1+\mu} - \varepsilon$, a contradiction. Thus, it must be placed on the left of $\mathcal{S}_0$. The position of this item has to be $-\frac{\mu}{1+\mu} - \varepsilon$. This yields the new center of gravity of $(k \cdot (\frac{1}{1+\mu} - \varepsilon) - \frac{\mu}{1+\mu} - \varepsilon)/k+1$. This center of gravity must be located to the right of $-\varepsilon$. Thus, we have

$$k \cdot (\frac{1}{1+\mu} - \varepsilon) - \frac{\mu}{1+\mu} - \varepsilon + (k+1) \cdot \varepsilon > 0 \quad \Leftrightarrow \quad k - \mu > 0 \quad \Leftrightarrow \quad k > \mu$$

Because we cannot stack $\mu + 1$ items, we cannot have any strategy achieving a deviation of $[0, \frac{1}{1+\mu} - \varepsilon]$. We conclude that our strategy given in Theorem 4 must be optimal.

Corollary 5. *With the given strategy for a uniform system where each item has length ℓ, the center of gravity deviates in $[0, \frac{\ell}{1+\mu}]$, which is optimal.*

4.2 Optimally Loading Without Stacking but With Minimal Space

Assume that the height of the ship to be loaded does not allow stacking items. This makes it necessary to ensure that the space consumption of the packing is minimal. We restrict ourselves to plane placements such that each state is connected. For simplicity, we assume w.l.o.g. that $\ell_1 \geq \dots \geq \ell_n$ holds. First we argue that $\Delta(p) \geq \frac{\ell_2}{4}$ holds for an arbitrary connected plane placement p of S. Subsequently we give an algorithm that realizes this lower bound.

A fundamental key for this subcase is that the center of gravity of a connected plane state is the midpoint of the induced overall interval.

Observation 1. *Let s be a plane state such that the union of the corresponding intervals is an interval $[a, b] \subset \mathbb{R}$. Then $C(s) = \frac{a+b}{2}$.*

Lemma 3. *For each plane placement p of S, we have $\Delta(p) \geq \frac{\ell_2}{4}$.*

Proof. Let p be an arbitrary plane placement of $S = \langle (I_1, 1), \dots, (I_n, 1) \rangle$, let $\langle s_0, s_1, \dots, s_n \rangle$ be the sequence of states that are induced by p, and let $i, j \in \{1, \dots, n\}$ be such that $I_i = |\ell_1|$ and $I_j = |\ell_1|$ hold. Observation 1 implies that $|C(s_{i-1}) - C(s_i)| = \frac{\ell_1}{4} \geq \frac{\ell_2}{4}$ and $|C(s_{j-1}) - C(s_j)| = \frac{\ell_2}{4}$. Let m_i and m_j be the midpoints of I_i and I_j. As the intervals I_i and I_j do not overlap, we conclude that $|m_i| \geq \frac{\ell_2}{2}$ or $|m_j| \geq \frac{\ell_2}{2}$ holds. W.l.o.g. assume that $|m_i| \geq \frac{\ell_2}{2}$ holds. This implies that $|C(s_{i-1})| \geq \frac{\ell_2}{4}$ or $|C(s_i)| \geq \frac{\ell_2}{4}$ holds. In both cases, we obtain $\Delta(p) \geq \frac{\ell_2}{4}$, concluding the proof.

Lemma 4. *We can compute a placement p of S such that $\Delta(p) \leq \frac{\ell_2}{4}$.*

Proof. The main idea is as follows. We remember $\ell_1 \geq \dots \geq \ell_n$ and place the items in that order. In particular, we choose the positions of the items such that $C(s_1) := -\frac{\ell_2}{4}$ and $C(s_2) := \frac{\ell_2}{4}$. The remaining intervals are placed alternating, adjacent to the left and to the right side of the previously placed intervals.

In order to show that $C(s_i) \in [-\frac{\ell_2}{4}, \frac{\ell_2}{4}]$ holds for all $i \in \{0, \ldots, n\}$, we prove by induction that $C(s_i) \in [C(s_{i-2}), C(s_{i-1})]$ holds for all odd $i \geq 3$ and $C(s_i) \in [C(s_{i-1}), C(s_{i-2})]$ for all even $i \geq 4$. As Observation 1 implies $C(s_1) = -\frac{\ell_2}{4}$ and $C(s_2) = \frac{\ell_2}{4}$, this concludes the proof.

Let $i \geq 3$ be odd. We have $|C(s_{i-2}) - C(s_{i-1})| = \frac{\ell_{i-1}}{2}$. This is lower bounded by $\frac{\ell_i}{2}$ because $\ell_i \leq \ell_{i-1}$. Furthermore, we know that $|C(s_{i-1}) - C(s_i)| = \frac{\ell_i}{2}$. This implies $C(s_i) \in [C(s_{i-2}), C(s_{i-1})]$. The argument for the case of even $i \geq 4$ is analogous.

The combination of Lemmas 3 and 4 implies that our approach for connected placements is optimal.

Corollary 6. *Given an arbitrary system, there is a polynomial-time algorithm for optimally loading a general set of items without stacking and under the constraint of minimal space consumption for all intermediate stages.*

4.3 Optimally Loading Exponentially Growing Items

Similar to the previous section, we consider plane placements. Now we consider the case in which the items have exponentially rising lengths. This case highlights the challenges of uneven lengths, in particular when the sizes are growing very rapidly; without special care, this can easily lead to strong deviation during the loading process. We show how the deviation can be minimized.

Theorem 6. *There is a polynomial-time algorithm for optimally loading a set of items with lengths growing exponentially by a factor $x \geq 2$ in increasing order w.r.t. to their lengths.*

Details can be found in the full version of this paper [7].

5 Conclusion

We have introduced a new family of problems that aim for balancing objects w.r.t. their center of gravity during loading and unloading these objects, and have provided hardness results and optimal or constant-factor approximation algorithms.

There are various related challenges. These include sequencing problems with multiple loading and unloading stops (which arise in vehicle routing or tour planning for container ships); variants in which items can be shifted in a continuous fashion; batch scenarios in which multiple items are loaded or unloaded at once (making it possible to maintain better balance, but also increasing the space of possible choices); and higher-dimensional variants, possibly with inhomogeneous space constraints. All these are left for future work.

Acknowledgements. We would like to thank anonymous reviewers for providing helpful comments and suggestions improving the presentation of this paper. J. Mitchell is partially supported by the National Science Foundation (CCF-1526406).

References

1. Amiouny, S.V., Bartholdi, J.J., Vate, J.H.V., Zhang, J.: Balanced loading. Oper. Res. **40**(2), 238–246 (1992)
2. Bischoff, E.E., Marriott, M.D.: A comparative evaluation of heuristics for container loading. Eur. J. Oper. Res. **44**(2), 267–276 (1990)
3. Bischoff, E.E., Ratcliff, M.: Issues in the development of approaches to container loading. Omega **23**(4), 377–390 (1995)
4. Christensen, S.G., Rousøe, D.M.: Container loading with multi-drop constraints. Int. Trans. Oper. Res. **16**(6), 727–743 (2009)
5. Davies, A.P., Bischoff, E.E.: Weight distribution considerations in container loading. Eur. J. Oper. Res. **114**(3), 509–527 (1999)
6. Fasano, G.: A MIP approach for some practical packing problems: balancing constraints and tetris-like items. 4OR **2**(2), 161–174 (2004)
7. Fekete, S.P., von Höveling, S., Mitchell, J.S.B., Rieck, C., Scheffer, C., Schmidt, A., Zuber, J.R.: Don't rock the boat: algorithms for balanced dynamic loading and unloading. CoRR, abs/1712.06498 (2017). http://arxiv.org/abs/1712.06498
8. Gehring, H., Bortfeldt, A.: A genetic algorithm for solving the container loading problem. Int. Trans. Oper. Res. **4**(5–6), 401–418 (1997)
9. Gehring, H., Menschner, K., Meyer, M.: A computer-based heuristic for packing pooled shipment containers. Eur. J. Oper. Res. **44**(2), 277–288 (1990)
10. Gilmore, P., Gomory, R.E.: Multistage cutting stock problems of two and more dimensions. Oper. Res. **13**(1), 94–120 (1965)
11. Limbourg, S., Schyns, M., Laporte, G.: Automatic aircraft cargo load planning. JORS **63**(9), 1271–1283 (2012)
12. Lurkin, V., Schyns, M.: The airline container loading problem with pickup and delivery. Eur. J. Oper. Res. **244**(3), 955–965 (2015)
13. Mathur, K.: An integer-programming-based heuristic for the balanced loading problem. Oper. Res. Lett. **22**(1), 19–25 (1998)
14. Mongeau, M., Bes, C.: Optimization of aircraft container loading. IEEE Trans. Aerosp. Electron. Syst. **39**(1), 140–150 (2003)
15. Øvstebø, B.O., Hvattum, L.M., Fagerholt, K.: Optimization of stowage plans for roro ships. Comput. Oper. Res. **38**(10), 1425–1434 (2011)
16. Øvstebø, B.O., Hvattum, L.M., Fagerholt, K.: Routing and scheduling of roro ships with stowage constraints. Transp. Res. Part C: Emerg. Technol. **19**(6), 1225–1242 (2011)
17. Paterson, M., Peres, Y., Thorup, M., Winkler, P., Zwick, U.: Maximum overhang. Am. Math. Mon. **116**(9), 763–787 (2009)
18. Paterson, M., Zwick, U.: Overhang. Am. Math. Mon. **116**(1), 19–44 (2009)
19. Pollaris, H., Braekers, K., Caris, A., Janssens, G.K., Limbourg, S.: Vehicle routing problems with loading constraints: state-of-the-art and future directions. OR Spectr. **37**(2), 297–330 (2015)
20. Sevastianov, S.: On some geometric methods in scheduling theory: a survey. Discret. Appl. Math. **55**(1), 59–82 (1994)
21. Sevastianov, S.: Nonstrict vector summationin multi-operation scheduling. Ann. Oper. Res. **83**, 179–212 (1998)
22. Souffriau, W., Demeester, P., Berghe, G.V., De Causmaecker, P.: The aircraft weight and balance problem. Proc. ORBEL **22**, 44–45 (1992)
23. Vancroonenburg, W., Verstichel, J., Tavernier, K., Berghe, G.V.: Automatic air cargo selection and weight balancing: a mixed integer programming approach. Transp. Res. Part E: Logist. Transp. Rev. **65**, 70–83 (2014)