Briskorn, Dirk; Boysen, Nils; Zey, Lennart

Article — Published Version

Scheduling of e-commerce packaging machines: blocking machines and their impact on the performance–waste tradeoff

Journal of Scheduling

Suggested Citation: Briskorn, Dirk; Boysen, Nils; Zey, Lennart (2024) : Scheduling of e-commerce packaging machines: blocking machines and their impact on the performance–waste tradeoff, Journal of Scheduling, ISSN 1099-1425, Springer US, New York, Vol. 28, Iss. 1, pp. 101-120, https://doi.org/10.1007/s10951-024-00826-9

This Version is available at:
https://hdl.handle.net/10419/323370

Journal of Scheduling (2025) 28:101–120
https://doi.org/10.1007/s10951-024-00826-9

Scheduling of e-commerce packaging machines: blocking machines and their impact on the performance–waste tradeoff

Dirk Briskorn[1] · **Nils Boysen**[2] · **Lennart Zey**[1]

Accepted: 1 October 2024 / Published online: 25 October 2024

Abstract

To streamline their fulfillment processes, many e-commerce retailers today use automated packaging machines for their outbound parcels. An important performance–waste tradeoff is associated with these machines: To reduce packaging waste when handling different sized goods, packaging machines should be able to handle different carton sizes. However, more carton sizes lead to a more involved scheduling process, so that the throughput performance deteriorates (and vice versa). To investigate this tradeoff, this paper develops scheduling procedures for a specific type of packaging machine, called *blocking machines*. These packaging machines provide multiple back-to-back packaging devices, each continuously processing a dedicated carton size, but blocking each other whenever incoming goods are not properly ordered according to carton sizes on the infeed conveyor. To reduce the resulting throughput loss, we derive various scheduling problems for optimizing the inflow of goods, provide a thorough analysis of the computational complexity, and derive an exact dynamic programming approach that is polynomial in the number of orders to be packed. This allows us to solve even large real-world instances to proven optimality with which we can analyze the performance–waste tradeoff of blocking machines.

Keywords E-commerce · Packaging machines · Environmental impact · Scheduling

1 Introduction

Because of the repetitive and physically demanding nature of warehouse work, many efforts have been made to reduce the burden on human workers. In addition to forklifts and conveyors, which have an even longer tradition, crane-operated high-bay warehouses, for example, have been assisting in the storage and retrieval of goods since the 1960 s (Boysen and de Koster, 2024). Driven by the huge success of e-commerce, the last decades have seen further progress in the field of warehouse automation (see Azadeh et al. 2019). Today, there are automated solutions for all basic warehousing functions (see Boysen et al. 2019): For example, there are mobile shelf-lifting robots for the transport function, autonomous mobile robots for picking rectangular goods from shelves, robotic arms with vacuum grippers for picking from bins, mobile robots with tiltable trays for order sorting, and industrial robots for palatalizing boxes and cartons.

This paper deals with the automation of the packaging function, where the picked (and consolidated) products required by customer orders are either wrapped in plastic film or packed in cardboard boxes by a fully automated packaging machine. Machines for the latter case, which we focus on in this paper, are very common in e-commerce because cardboard boxes provide better protection, especially for fragile goods (Escursell et al., 2021). These machines are fed with goods and cardboard packaging material via a conveyor and a feeding shaft, respectively. Most machines include a cutting mechanism to reduce the size of the boxes to fit the orders. This avoids higher postage charges and excess fill material caused by unnecessarily large packages. The packaging material is then folded, sealed and labeled by the machine. Finally, the finished boxes are conveyed to the shipping area.

✉ Nils Boysen
nils.boysen@uni-jena.de
http://www.om.uni-jena.de

Dirk Briskorn
briskorn@uni-wuppertal.de
http://www.prodlog.uni-wuppertal.de

Lennart Zey
zey@uni-wuppertal.de

[1] Bergische Universität Wuppertal, Professur für BWL, insbesondere Produktion und Logistik, Rainer-Gruenter-Str. 21, 42119 Wuppertal, Germany

[2] Friedrich-Schiller-Universität Jena, Lehrstuhl für Operations Management, Carl-Zeiß-Straße 3, 07743 Jena, Germany

Fig. 1 Paper E-Com Fit of Hugo Beck with two back-to-back packaging devices (Source: Hugo Beck)

In their packaging machine evaluation paper, Pfoser et al. (2021) report state-of-the-art throughput performance of up to 1000 packages per hour for single-piece orders. For multi-piece orders, throughput is lower because their variety is still a greater technological challenge. However, since the majority of e-commerce orders are for a single item (the average number of items ordered at Amazon Germany, for example, is only 1.6, see Boysen et al. 2019), automated packaging machines are very common in today's e-commerce fulfillment centers.

There are several types of packaging machines, which we compare in more detail in Sect. 2. One common setup that we will focus on in this paper is the *blocking machine*, shown in Fig. 1. To avoid the disadvantages of setups where cartons must be changed on-the-fly whenever a different carton size is required, blocking machines use multiple downstream packaging devices, each of which is permanently associated with a specific carton size. These subsequent packaging devices are arranged back-to-back along the infeed conveyor, causing blockages whenever incoming goods are not properly ordered by size. Because goods cannot overtake each other on the infeed conveyor, a good that is assigned to an upstream packaging device will block subsequent goods that require a downstream device. Thus, one or multiple packaging slots in a packaging batch (i.e., the set of orders concurrently packed by all parallel packaging devices) remain empty and a *blocking loss* occurs. A suitable sequencing of orders within the infeed sequence can reduce the blocking loss and thus improve the throughput performance of a blocking machine. This directly leads to the following performance–waste tradeoff: More carton sizes to choose from reduces packaging waste, but also tends to increase the loss of throughput performance due to blockings. To explore the performance–waste tradeoff, this paper provides the following contributions:

- We introduce a novel scheduling problem for blocking machines that minimizes blocking loss for different inflows of orders. For example, many e-commerce retailers equip their human pickers with multi-bin picking carts to collect completed orders in a sort-while-pick picking process (e.g., see De Koster et al. 2007). In this case, the inflow to a packing machine can be altered by changing the order in which these carts are processed and the order in which each cart's orders are placed on the machine's infeed conveyor. We describe alternative inflows and their effect on the scheduling problem in Sect. 3.
- We provide a thorough analysis of computational complexity for all resulting problem variants. This leads to an exact dynamic programming (DP) algorithm that is polynomial in the number of goods to be packed, once the number of carton sizes is fixed. This allows us to solve large instances of real-world size within very short runtimes to proven optimality.
- With this algorithm in hand, we explore the performance–waste tradeoff of packaging machines. For a given number of carton sizes to be provided, we first optimize a suitable selection of specific carton sizes to minimize packaging waste. Then, we minimize the blocking loss by solving our scheduling problem for the selected carton sizes. This allows us to quantify packaging waste and throughput loss if different numbers of carton sizes are to be provided. This delivers decision support for warehouse managers who need to make the right choice of carton sizes.

The rest of the paper is organized as follows. In Sect. 2, we discuss related decision problems and review the literature. In Sect. 3, we define the different variants of the packaging machine scheduling problem (PMSP) treated in this paper, which differ in the inflow of goods and the flexibility to reduce blocking loss by changing the processing sequence of orders. Section 4 contains a detailed analysis of the computational complexity, and Sect. 5 provides an efficient exact solution method based on DP. In Sect. 6 and Sect. 7 we elaborate on our computational study and explore the performance–waste tradeoff. Finally, Sect. 8 concludes the paper.

2 Related decisions and literature review

A recent in-depth survey paper on sustainability in e-commerce packaging (Escursell et al., 2021) and our own (thorough) literature search reveal that there is no previous research on packaging machine scheduling and its impact on the waste–performance tradeoff. However, in order to position our work in relation to previous research, we take a look at related decision tasks. Specifically, we address (i) the choice between different packaging machine setups, (ii) the choice of carton sizes, and (iii) the packing of goods into car-

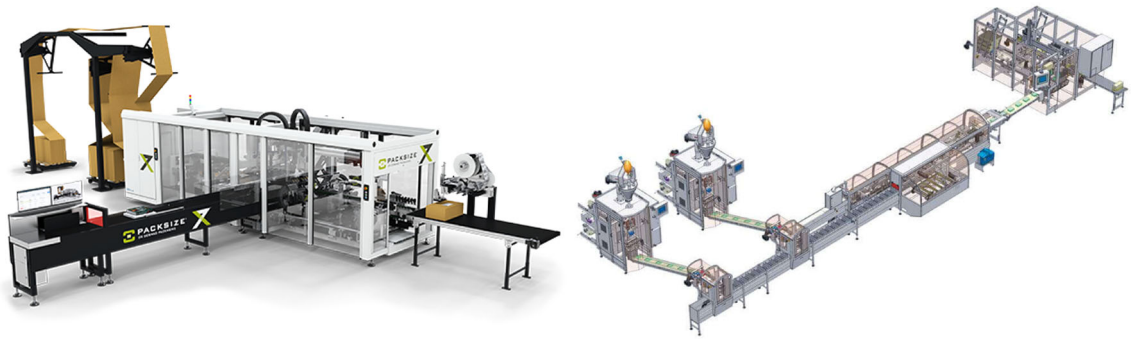


Fig. 2 Alternatives to blocking machines. Left: Setup machine X7^{TM} of Packsize requiring setups for carton size switches (Source: Packsize). Right: Parallel packing machines connected by a conveyor system (Source: Rovema)

tons. Finally, we also discuss (iv) other scheduling problems with a similar problem structure.

(i) *Choice of packaging machine setups:* Based on numerous site visits to e-commerce warehouses, a thorough evaluation of packaging machine manufacturers' websites, and discussions with managers and consultants in the field, the authors are aware of two alternatives to blocking machines. Setup machines (see Fig. 2(left)) are equipped with an automated carton switching device. They automatically load the currently required carton size into the carton feeding shaft of their single packaging device and remove the old one. The resulting setups cannibalize packaging capacity and, thus, also create a waste–performance tradeoff. Since high-speed setups are mandatory for economical application of automated packaging, we were told that they are a common source of errors that require a lot of machine maintenance. On the positive side, setup machines do not require redundant hardware. To completely avoid the waste–performance tradeoff, multiple independent packaging machines, each dedicated to a specific carton size, can be used in parallel. These machines are accessed via switches from the main conveyor (see Fig. 2(right)). Independent machines without carton size switches, however, require a high investment. Blocking machines can be seen as a compromise between these two alternatives. They use dedicated packaging devices without setups, but allow reuse of parts of the hardware. The price for this is a blocking loss if products and their demanded carton sizes are not properly sequenced in the infeed. In our research, we only address blocking machines and leave a more comprehensive evaluation of all different setups to future research.

(ii) *Choice of carton sizes:* To achieve economies of scale in purchasing and to limit handling effort, retailers cannot provide a perfectly fitting carton for every product (or multi-piece order). Therefore, the number of carton sizes used is reduced to a few dozen at most. Typically, automated packaging demands an even smaller portfolio of carton sizes than manual packaging due to the higher flexibility of human work. Once the number of different carton sizes to be used is determined, choosing the specific sizes that minimize packaging waste for a given set of orders an optimization problem by itself. Heuristics based on clustering methods (Liu et al., 2013; Brinker and Gündüz, 2016) and genetic algorithms (Singh and Ardjmand, 2020) have been introduced. For our operational scheduling problem, we assume that the set of available carton sizes has already been decided in a previous (long-term) decision task. However, for our evaluation of the waste–performance tradeoff, we also determine the minimum waste carton sizes for a given set of orders to be packed (see Sect. 7.1). To do so, we apply a straightforward DP approach, similar to the one proposed by Lee et al. (2015), which optimizes the height of crates in the chemical industry to accommodate a given set of goods.

(iii) *Packing of goods into cartons:* The question of how to pack the goods of multi-piece orders into boxes is closely related to the bin packing problem. Surveys on this classic of the operations research domain are provided, for example, by Delorme et al. (2016) (exact algorithms) and Coffman et al. (2013) (heuristics). However, unlike bin packing, where all bins are the same size and the number of bins to pack all products must be minimized, e-commerce retailers have boxes of different sizes. Their choice is to determine the best box for each order jointly with the packing task, minimizing packaging waste and add-on volume. Fontaine and Minner (2023) proposes an efficient branch-and-repair method for this decision. For our scheduling problem, we assume that the choice of carton size into which each order is to be packed is already given. Recall that most orders for which automated packaging is applied are single-piece orders anyway, where the choice of the best-fitting carton size is trivial.

(iv) *Related scheduling problems:* There is scheduling research for manufacturers of packaging machinery (e.g., (Adler et

al., 1993)) and for manufacturers of packaging materials (e.g., (Li et al., 2018)). However, we are not aware of any scientific scheduling research that specifically addresses e-commerce retailers and their use of packaging machinery to package their own goods. For setup machines, minimizing setup loss for a given order set on a single machine is a special case of the well-known traveling salesman problem (Burkard et al., 1998). For blocking machines, where the sizes of goods of each packaging batch must be properly ordered according to the subsequent packaging devices, such that the total number of packaging batches is minimized, this transformation is not available. Furthermore, unlike traditional machine scheduling, where the sequence in which orders are processed is typically unrestricted, e-commerce packaging machines are involved in a multi-stage process (e.g., including picking and order consolidation (Boysen et al., 2019)). The resulting material flow between these stages often relies on a batchwise transport (see Sect. 3.1), so the flexibility to sequence the inflow of goods is limited (i.e., only the sequence of batches and the order sequence per batch can be changed). Extensions of traditional machine scheduling to account for such limited flexibility in order sequencing are known as *group technology*. Machine scheduling adaptations for this type of inflow have been studied, for example, by Ng et al. (2005), Janiak et al. (2005), and Li et al. (2011). However, due to our completely different objective function these previous scheduling problems are not directly applicable.

We conclude that our PMSP and its influence on the waste–performance tradeoff of packaging machines has not been addressed before.

3 Problem description

To ease understanding of the variants of PMSP treated in this paper, we start with a verbal problem characterization, examples, and the discussion of our basic assumptions in Sect. 3.1. Afterward, we provide a precise mathematical problem definition in Sect. 3.2.

3.1 Problem characterization, examples, and assumptions

The basic decision task of the PMSP is the sequence in which orders to be processed by the packaging machine are placed onto the machine's infeed conveyor. Each order demands a specific carton size to be properly packed, which are provided by multiple subsequent packaging devices, arranged back-to-back along the conveyor. Since orders cannot overtake each other on the conveyor, it may occur that orders block each other. A blocking occurs whenever a preceding order demands a carton size provided by an anterior (or the same) packaging device, so that a subsequent order cannot reach its dedicated, yet blocked position along the belt. Thus, one or multiple packaging slots in a packaging batch (i.e., the set of orders concurrently packed by all parallel packaging devices) remain empty and a *blocking loss* occurs. A suitable sequencing of orders within the infeed sequence can reduce the blocking loss and thus improve the throughput performance of a blocking machine. The PMSP aims to minimize the number of packaging batches that are required to process all orders.

Since packing is part of a multi-stage order fulfillment process, we are typically not completely free in the sequence orders are loaded into the packaging machine's infeed sequence. Depending on the type of inbound stream, we face different levels of flexibility how to manipulate the infeed sequence. Many warehouses apply multi-bin picking carts, as depicted in Fig. 3. Such a picking cart either accompanies a human picker during a picker-to-parts process. Each bin of a cart is then associated with a specific customer order, so that the picker can directly place each demanded product into the right customer bin in a sort-while-pick process (De Koster et al., 2007). Alternatively, picking carts are also applied to deliver orders from the consolidation stage, where products get sorted after picking in a sort-after-pick process (Boysen et al., 2019). In both cases, orders (each stored in a separate bin) arrive in picking carts at the infeed station of the packaging machine, where order after order is placed onto the infeed conveyor (either manually or by an automated solution). Based on this basic process, we differentiate the following types of inbound streams:

- (a) *Given cart sequence:* The sequence in which the picking carts are processed at the infeed station can be given. This is, for instance, the result when processing the carts after the widespread first-come-first-served rule. Hence, only the sequence in which the orders of each subsequent cart are placed onto the conveyor is the lever to alter the packaging machine's infeed sequence. Note that each cart must be completely processed before the next one can be started.
- (b) *Arbitrary cart sequence:* Alternatively, next to the order sequence per cart also the sequence in which a given set of picking carts, waiting at the loading station, are processed can be part of the decision. Once the cart sequence is determined, again, each cart must be completely processed before the next one can be started. Nevertheless, this leaves more flexibility to improve the infeed sequence.
- (c) *Arbitrary infeed sequence:* Finally, the largest flexibility is at hand if all orders of the given order set can

Fig. 3 Picking carts in a warehouse (Source: Lightning Pick)

be brought into an arbitrary infeed sequence. This case arises if the current planning run only has to decide on the order sequence of a single picking cart or if incoming goods have been intermediately stored in a random access buffer (e.g., in an ASRS, see Boysen and Stephan, 2016).

Note that other warehouses do not apply picking carts to deliver orders to packaging machines. If only single-piece orders are picked, then all demanded products can be placed into the same large bin. Order consolidation is not necessary, because it is known that each product refers to its own order. These bins with single-piece orders can also arrive at the infeed station, e.g., delivered by forklifts, AGVs, or on a conveyor. In relation to the cart-based process elaborated above, bins correspond to carts and products to orders. Although the physical process is different, it can be modeled by the same three types of inbound streams as elaborated above. Note, furthermore, that a fixed and given infeed sequence of orders, which cannot be altered (e.g., because it directly arrives from the picking area on a conveyor), leaves no optimization problem and is thus not considered in our problem differentiation. These alternative inbound streams offer different levels of flexibility to alter the infeed sequence. Exploring the impact of these flexibility levels (also including a fixed infeed sequence) on the waste–performance tradeoff is part of our computational study in Sect. 7.

Example 1 The basic input data of the example instance depicted in Fig. 4a are four picking carts, denoted A, B, C, and D, each filled with two different orders. Each order's demanded carton size is given by the white number within the respective gray order square. We have demanded carton sizes from 1 to 5, so that our packaging machine also has five back-to-back packaging devices each servicing one of the sizes. On the right side, we see two different infeed sequences (b) and (c). A given cart sequence leads to three packaging batches and a blocking loss of seven (see solution (b)). The relationship among these two performance measures is as follows: We have three batches with five packaging devices. This leads to 15 slots, among which eight are used by the given orders, whereas seven remain unused and constitute blocking loss. If the cart sequence is part of the decision, then optimal solution (c) leads to only two packaging batches and a blocking loss of $2 \cdot 5 - 8 = 2$.

Example 2 Now, we consider the same situation as in Example 1, where, however, the managerial decision has been made to merely provide two carton sizes of size 3 and 5. Figure 5d indicates the modified input data, where the actual sizes of the orders, which are equal to the sizes of Example 1, are indicated by the white subscripts within the gray order squares. Their assignment to the next larger available carton size is given by the (normal-sized) white numbers in the order squares. This induces a packaging waste of six, due to putting orders into cartons of larger size than is actually required. An optimal solution of PMSP, which does not improve if the cart sequence can be altered, is depicted in (e). Because we only have two carton sizes, we only require two back-to-back packaging devices, which reduces the investment cost for the blocking machine. This, however, also implies less packaging capacity, so that we need five packaging batches (and thus have to accept a longer makespan until all orders are completed). Two packaging devices and five batches directly imply a blocking loss of $2 \cdot 5 - 8 = 2$ slots that remain empty, which is however no improvement compared to solution (c) of Example 1. We can deduce two issues for these two examples: (i) More sequencing flexibility (i.e., if the cart sequence is part of the decision and not given) as well as (ii) fewer carton sizes can but need not reduce the throughput loss.

Before, we continue with a precise problem definition of our PMSP variants, we discuss the (simplifying) assumptions made in this paper to derive the PMSP in its very basic form:

- We consider only a single packaging machine with fixed order assignments. When multiple parallel packaging machines are available, the order assignment is another relevant decision. Our solution methods can be applied to evaluate different order-machine assignments, but we leave the evaluation of such a decomposition approach to future research.

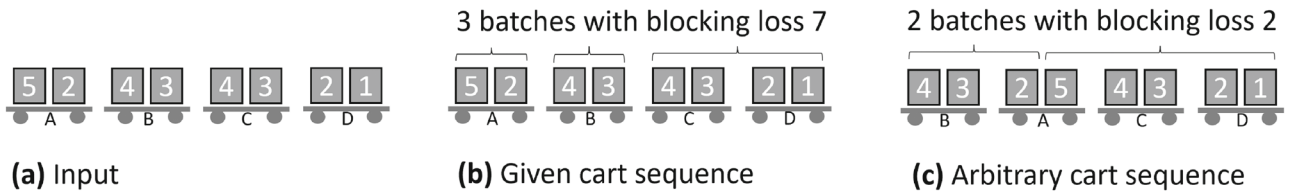


Fig. 4 Example 1 of PMSP with and without given cart sequence

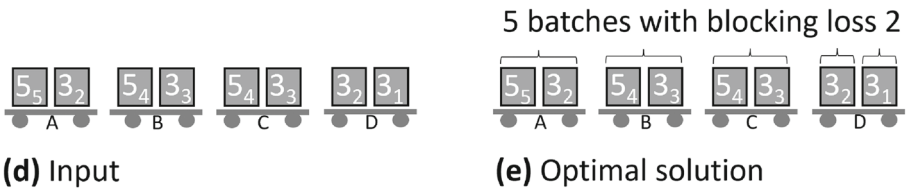


Fig. 5 Example 2 of PMSP with fewer carton sizes

- For convenience, we neglect the potential impact of previous planning runs. That is, we neglect the possibility that the last orders of the preceding planning run could potentially share a packing batch with the first orders of the subsequent planning run. It is easy to relax this simplification, but we have chosen to stick with the simplest problem setup.
- We assume that variable-speed conveyor segments ensure that orders arrive in the infeed sequence without gaps. Of course, the scheduling of packaging machines is particularly important when packaging is a bottleneck stage. In this case, gaps degrade the throughput of a bottleneck resource. Therefore, we assume that a retailer interested in PMSP has already eliminated this obvious source of wasted bottleneck capacity. If gaps exist, they could result in additional blocking loss despite properly sequenced orders approaching a blocking machine. We leave the consideration of gaps to future research.
- We assume that each order requires a specific carton size. The additional flexibility provided by hierarchical compatibility, which allows orders to be packed in larger cartons than necessary to reduce throughput loss at the cost of additional waste, is thus neglected and left for future research.
- Finally, we assume that all input data is known with certainty. Note that for a reliable automated packaging process, detailed size information about the arriving products must be available anyway, so this assumption does not seem to be a severe constraint in our case.

Given this decision context, our PMSP is precisely defined in the following section.

3.2 Problem definition

We consider a given set J of orders, where each order $j \in J$ has a (carton) size $s_j \in \{1, \ldots, S\}$ and a cart (number) $c_j \in \{1, \ldots, C\}$. Here, S is the number of distinct sizes and C is the number of carts. We assume that each size in $\{1, \ldots, S\}$ and each cart in $\{1, \ldots, C\}$ is related to at least one order (otherwise, we can reduce the number of sizes or the number of carts). Note that both, S and C, are implied by J and, thus, are given, as well. We denote the set of orders with cart number c as J_c and will say that order j is in cart c if $j \in J_c$. A solution is a permutation σ of orders in J. We denote the k-th order in σ by $\sigma(k)$. A solution σ is feasible if and only if for each pair of carts c and c' with $c < c'$ either all orders with cart number c precede all orders with cart number c' in σ or the other way around (if cart sequencing is part of the decision). Feasibility of a solution σ, thus, implies that orders appear clustered by cart numbers in σ.

To evaluate a solution, we denote by $\Phi(\sigma) = |\{k \mid k = 1, \ldots, |J| - 1, s_{\sigma(k)} \leq s_{\sigma(k+1)}\}|$ the number of packaging batches of a solution, which equals the number of orders followed immediately by an order of larger or equal size in σ). We say that there is a *break* between positions k and $k+1$, if $s_{\sigma(k)} \leq s_{\sigma(k+1)}$, that is, if a new batch starts in position $k+1$. Because each packaging batch takes constant time for packing the orders at the subsequent packaging devices in parallel, we, then, associate $\Phi(\sigma)$ with the makespan, that is,

$$C_{\max}(\sigma) = \Phi(\sigma).$$

The PSMP is to determine a solution that minimizes $C_{\max}(\sigma)$ among all feasible solutions.

While the special case of PSMP with $C = 1$ covers the setting with an arbitrary infeed sequence (see Sect. 3.1), the setting with a given infeed sequence is not covered by a special case of PSMP. Therefore, we define a variant of PSMP,

namely PSMP-fixed, to formalize this problem. PSMP-fixed has the same input as PSMP, the same solution space, and the same evaluation of solutions. A solution is considered feasible, however, if and only if for each pair of carts c and c' with $c < c'$ all orders with cart number c precede all orders with cart number c' in σ. We, thus, require that the given cart sequence has carts in increasing order of their numbers. The PSMP-fixed is to determine a solution that minimizes $C_{\max}(\sigma)$ among all feasible solutions.

4 Analysis of computational complexity

This section provides an in-depth analysis of computational complexity for PSMP and PSMP-fixed. For the former problem, we distinguish cases where the given number of carts is either $C = 1$, C is fixed to a given constant, or C is part of the input. For both problems, the given number of carton sizes S can either be fixed or is part of the input. Thus, we consider eight problem variants in total.

We start with PSMP-fixed. We consider the greedy style algorithm sketched in the following. We assign orders in J one by one to positions in the permutation in ascending order. For each position k, an order is assignable if it has not been assigned to a previous position yet and its cart number refers to the cart currently processed according to the given sequence. Let s be the size of the order in position $k-1$ if $k > 1$, and $s = \infty$ otherwise. Let, furthermore, J' be the set of available orders with sizes smaller than s. If $J' = \emptyset$, then we choose an order with maximum carton size among all available orders for position k. This implies a break between positions $k-1$ and k, if $k > 1$. If $J' \neq \emptyset$, then we choose an order with maximum size among orders in J' for position k. This implies no break between positions $k-1$ and k, if $k > 1$. We refer to this algorithm as GREEDY and introduce the following lemma related to it.

Lemma 1 *GREEDY achieves an optimum solution for PSMP-fixed.*

Proof Consider an optimum solution σ^* and a further solution σ obtained by GREEDY. Let $k < |J|$ be the first position where σ^* and σ differ. Note that if there is no such position σ is optimum. Obviously, $c_{\sigma^*(k)} = c_{\sigma(k)}$. We distinguish two cases regarding the sizes of $\sigma^*(k)$ and $\sigma(k)$.

- If $s_{\sigma^*(k)} = s_{\sigma(k)}$, then we can simply switch $\sigma^*(k)$ and $\sigma(k)$ in σ^* and obtain an optimum solution that equals σ up to position k.
- If $s_{\sigma^*(k)} \neq s_{\sigma(k)}$, we modify σ^* as follows. Let k' be the position where $\sigma(k)$ is assigned to according to σ^*, that is $\sigma^*(k') = \sigma(k)$. Note that $k' > k$. We move $\sigma^*(k')$ to position k and delay all orders in positions k to $k'-1$ by one position. We refer to the modified solution as σ'.

  Note that positions 1 to $k-1$ and $k'+1$ to $|J|$ are not modified. Furthermore, between positions $k+2$ and k' orders in σ' are in the same relative order as in σ^*. Hence, the total number of breaks immediately before the jobs in these positions is not higher in σ' than in σ^*. For the consideration of positions k and $k+1$ we distinguish two cases in the following

  – If $s_{\sigma^*(k)} > s_{\sigma(k)}$, then a break occurs in σ^* between $k-1$ and k but no break occurs in σ between $k-1$ and k, because $s_{\sigma(k)}$ is maximum among orders in J'. This holds true if $J' \neq \emptyset$ and among all available orders if $J' = \emptyset$. Hence, there is no break in σ' between $k-1$ and k. However, there is a break in σ' between k and $k+1$, because there is one in σ^* between $k-1$ and k and $s_{\sigma'(k)} < s_{\sigma^*(k)}$. So, the total number of breaks between $k-1$ and k and between k and $k+1$ is 1 in both, σ^* and σ'.
  – If $s_{\sigma^*(k)} < s_{\sigma(k)}$, then a break occurs in σ^* between $k-1$ and k and a break occurs in σ between $k-1$ and k, because GREEDY arranges a break only if $J' = \emptyset$. Then, there is a break in σ' between $k-1$ and k because there is one in σ. There is no break in σ' between k and $k+1$, because $s_{\sigma'(k)} > s_{\sigma^*(k)}$. Hence, again the total number of breaks between $k-1$ and k and between k and $k+1$ is 1 in both, σ^* and σ'.

  Hence, in both cases σ' is optimum, as well.

Concluding, there is an optimum solution coinciding with σ up to position k and, by applying the argument in an iterative manner, σ is optimum. □

Now, we are able to easily verify that the infeed sequence in Fig. 4b is indeed optimum for given cart sequence ⟨A,B,C,D], because it is the output of GREEDY. Regarding the computational complexity, the following theorem follows.

Theorem 1 *PSMP with a fixed number of carts and PSMP-fixed can be solved in polynomial time.*

Proof We can evaluate each sequence of carts using GREEDY according to Lemma 1. It is easy to see that GREEDY runs in polynomial time and there is a fixed number $C!$ of cart sequences and, hence, this procedure runs in polynomial time. □

Recall that the case $C = 1$ corresponds to an arbitrary infeed sequence as mentioned in Sect. 3.1. Hence, PSMP with an arbitrary infeed sequence can be solved in polynomial time. Finally, we consider PSMP with a fixed number of carton sizes S. We start with the basic idea of the approach, which is to separate the decisions about breaks between orders of the same cart and those of consecutive carts in a solution. We refer to these types as *internal breaks* and *external breaks*. We

do so, by guessing the number $C_{s,s'}$ of carts with first order's size s and last order's size s' for each pair (s, s') of sizes in an optimum solution. We refer to a set of such numbers (one number for each pair (s, s')) as a size profile.

Example 1 (cont.): The size profiles corresponding to the solution depicted in Fig. 4b and c, respectively, are specified as

$$\begin{pmatrix} 0&0&0&0&0\\ 1&0&0&0&0\\ 0&0&0&0&0\\ 0&0&2&0&0\\ 0&1&0&0&0 \end{pmatrix} \text{ and } \begin{pmatrix} 0&0&0&0&0\\ 1&0&0&0&1\\ 0&0&0&0&0\\ 0&0&2&0&0\\ 0&0&0&0&0 \end{pmatrix}.$$

For example, we have $C_{4,3} = 2$, because in both solutions there are two carts with first order's size 4 and last order's size 3. However, while the solution in Fig. 4b has a cart with first order's size 5 and last order's size 2 and, thus, $C_{5,2} = 1$, the solution in Fig. 4c has no such cart. Thus, we have $C_{5,2} = 0$. We determine (i) the sequence of orders within each cart, such that there are exactly $C_{s,s'}$ carts with first order's size s and last order's size s' and (ii) the sequence of carts, then, for each size profile. Note that both decisions together imply a feasible solution. Note, furthermore, that for taking the second decision only first order's size and last order's size of a cart are relevant. Hence, we can take the first decision irrespective of the second.

The procedure, which we dub SIZE_PROFILE, enumerates all size profiles and determines (i) the sequence of orders within each cart and (ii) the sequence of carts as detailed below. We restrict ourselves to size profiles with $\sum_{(s,s')\in S\times S} C_{s,s'} = C$.

1. To determine the sequence of orders within the each cart, we first determine the minimum number of internal breaks $\Delta_{c,s,s'}$ between orders of cart c, if the first order's size is s and the last orders size is s'. We can use a straightforward adaption of GREEDY with $C = 1$, where we have no freedom to decide the first and the last position and the corresponding orders are eliminated from the set of available orders. We refrain from giving a formal proof, because it is essentially the same as the proof for Lemma 1 (note that additionally $k' < |J|$ for the variant at hand). Hence, we can determine all $\Delta_{c,s,s'}$ values in polynomial time.

   Having determined $\Delta_{c,s,s'}$ for each cart c and pair of sizes (s, s'), we now consider a bipartite graph $G = (V, U, E)$ where nodes in $V = \{1, \ldots, C\}$ correspond to carts and nodes in U correspond to pairs of sizes. Exactly $C_{s,s'}$ nodes in U correspond to pair (s, s'). An edge between node $c \in V$ and a node in U corresponding to the pair of sizes (s, s') reflects cart c to have first order's size s and last order's size s' (and exists only if c can have first order's size s and last order's size s'). Choosing this edge in the following corresponds to choosing the sequence of orders for cart c implying $\Delta_{c,s,s'}$ internal breaks determined by the adaption of GREEDY. Consequently, an edge between $c \in V$ and a node in U corresponding to pair of sizes (s, s') has weight $\Delta_{c,s,s'}$, and we determine a minimum weight perfect matching G. Such a matching implies a choice of a sequence of orders for each cart, which is in line with the given size profile and has a minimum number of internal breaks between orders of the same cart.

   Example 1 (cont.): Table 1 outlines $\Delta_{c,s,s'}$ for each cart c and each pair (s, s') of sizes of the instance depicted in Fig. 4a. A numerical value is given only if c can have first order's size s and last order's size s'. Moreover, Fig. 6 depicts bipartite graphs and highlights minimum weight perfect matchings for two size profiles of this instance instance.

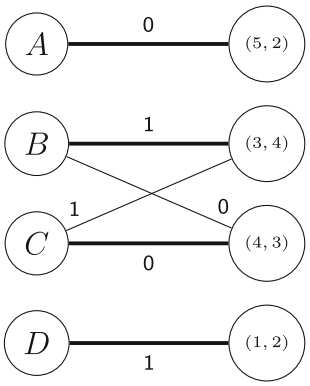
Table 1 $\Delta_{c,s,s'}$ for each cart c and pair of sizes (s, s') in Example 1

(s, s')	A	B	C	D
(1, 2)	–	–	–	1
(2, 1)	–	–	–	0
(3, 4)	–	1	1	–
(4, 3)	–	0	0	–
(2, 5)	1	–	–	–
(5, 2)	0	–	–	–

2. To determine the sequence of carts, we propose a DP approach, where we construct the sequence by adding carts one by one to an existing sequence of carts. A state $(\vec{k}, l)$ specifies the number $k_{s,s'}$ of carts scheduled so far with first order's size s and last order's size s' for each pair (s, s') and the size l of the last order in the sequence. We have a transition from state $(\vec{k}, l)$ to state $(\vec{k'}, l')$, if and only if exactly one number $k'_{s,s'}$ is increased by one as compared to $k_{s,s'}$, the others are identical, and the last size indicator $l' = s'$ reflects that a cart corresponding to pair (s, s') has been added to the sequence. Transition costs are in $\{0, 1\}$ and reflect whether or not an external break occurs before the first order of the newly added cart. Note that this is implied by l and the first order's size s. Note, furthermore, that the DP approach does not differentiate carts with the same pair of first order's size and last order's size (as determined above). Each state is evaluated by the number of external breaks between carts so far.

Theorem 2 *PSMP can be solved in polynomial time, if the number of carton sizes is fixed.*

Proof SIZE_PROFILE evaluates each size profile following the above points 1. and 2. The number of size profiles is in $O(C^{S^2})$ and, thus, polynomial (for fixed S). For each size



(a) Graph $G = (V, U, E)$ and minimum weight perfect matching (bold edges) for $C_{1,2} = C_{3,4} = C_{4,3} = C_{5,2} = 1$

(b) Graph $G = (V, U, E)$ and minimum weight perfect matching (bold edges) for $C_{2,1} = C_{2,5} = 1$ and $C_{4,3} = 2$

Fig. 6 Examples for bipartite graph and minimum weight perfect matching

profile, first, $\Delta_{c,s,s'}$ for each cart c and pair of sizes (s, s') is determined by running the adaption of GREEDY $O(CS^2)$ times. Recall that GREEDY runs in polynomial time according to Lemma 1. Second, a minimum weight perfect matching in bipartite graph G is determined, which can also be done in polynomial time. Third, in the above DP approach we have $O(SC^{S^2})$ states and $O(S^2)$ transitions starting from each of them. Each transition can be evaluated in $O(1)$ time and, thus, the overall complexity is $O(S^3C^{S^2})$ and, thus, polynomial (for fixed S). □

The above complexity results (and the open case, we were not able to resolve in this paper) are summarized in Table 2. We draw the following conclusion from our analysis of computational complexity. The long-term decision for the number of carton sizes S is mandatory to setup a packaging machine and is thus already made when the operational scheduling decisions targeted in this paper arise. Thus, the number of available carton sizes is fixed from the perspective of operational scheduling, and we can solve our PSMP in polynomial time. What is even more, the typical number of available carton sizes in the warehouses we have visited has never exceeded a handful and according to our contacts to packaging machine manufacturers is definitely below a dozen. Hence, our PSMP can efficiently be solved to optimality in real-world warehouses. This, however, requires an evaluation of all cart sequences (see Theorem 1) and of all pairs of size profiles (see Theorem 2), which can still take some time for real-world settings. In the following section, we streamline this optimization task significantly be introducing a DP procedure.

5 Solving PSMP

In this section, we propose solution methods for PSMP. We start by developing optimality properties in Sect. 5.1, which are then exploited in the DP approach presented in Sect. 5.2.

5.1 Properties of optimal solutions

First, we focus on the sequences of orders of the same cart. We will see that only a small fraction of all orders needs to be considered in most cases. Let Δ_c be the minimum number of internal breaks we can achieve for cart c.

Lemma 2 *There is an optimum solution, where*

- *the number of internal breaks is at most* $\Delta_c + 1$ *for each cart c,*
- *the first and the last cart c, respectively, has* Δ_c *internal breaks, and*
- *for each cart c with* $\Delta_c + 1$ *internal breaks, there is no external break between c and its predecessor cart or its successor cart.*

Proof We address the bullet points in the lemma one by one.

- Consider an optimum solution and a cart c with more than $\Delta_c + 1$ internal breaks. By choosing a sequence of orders of c with Δ_c internal breaks instead and keeping all other orders in their positions, we reduce the number of internal breaks by at least 2 and increase the number of external breaks by at most 2 (potentially adding external breaks between c and its predecessor cart and between c and its successor). Hence, the resulting solution is optimum, as well.
- Consider an optimum solution where the first (or the last) cart c has more than Δ_c internal breaks. By choosing

Table 2 Computational complexity of PSMP and PSMP-fixed

		S fix	S Part of the input
PSMP	$C = 1$	In $\mathcal{P}$ (Theorems 1 and 2)	In $\mathcal{P}$ (Theorem 1)
	C fix	In $\mathcal{P}$ (Theorems 1 and 2)	In $\mathcal{P}$ (Theorem 1)
	C part of the input	In $\mathcal{P}$ (Theorem 2)	Open
PSMP-fixed		In $\mathcal{P}$ (Theorem 1)	In $\mathcal{P}$ (Theorem 1)

a sequence of orders of c with Δ_c internal breaks and keeping all other orders in their positions, we reduce the number of internal breaks by at least 1 and add at most one external break between c and its successor (or between c and its predecessor). Hence, the resulting solution is optimum, as well.

- Consider an optimum solution and a cart c with $\Delta_c + 1$ internal breaks as well as an external break between c and its predecessor cart (or an external break between c and its successor cart). By choosing a sequence of orders of c with Δ_c internal breaks and keeping all other orders in their positions, we reduce the number of internal breaks by at least 1 and add at most one external break between c and its successor cart (or between c and its predecessor cart). Hence, the resulting solution is optimum, as well.

□

After establishing that we can restrict ourselves to sequences of orders for each cart with a small number of internal breaks, we aim at reducing the number of orders we need to consider explicitly. Let $n_{s,c}$ be the number of orders in cart c with size s and let $n_c^{\max} = \max\{n_{s,c} \mid s = 1, \ldots, S\}\}$. We define the *reduced cart* $\bar{c}$ corresponding to c, which contains those orders in J_c with a size that is required $n_c^{\max}$ times by cart c. Hence, $\bar{c}$ contains the subset $\bar{J}_c \subseteq J_c$ of orders in c that have one of the sizes most common in c. We refer to a size required by an order $\bar{J}_c$ as a *popular size* of c and denote the largest popular size of c by s_c^*. We will see in the following that there is a close relation between cart c and its reduced cart $\bar{c}$. Therefore, we can restrict ourselves to consider reduced carts only.

Example 3 We consider two carts 1 and 2 with $J_1 = \{j_1^1, \ldots, j_6^1\}$ and $J_2 = \{j_1^2, \ldots, j_9^2\}$. For simplicity reasons, we identify each order by its size only, that is we say $J_1 = \{1, 2, 2, 2, 3, 3\}$ and $J_2 = \{4, 5, 5, 6, 6, 6, 7, 7, 7\}$. For the reduced carts $\bar{1}$ and $\bar{2}$, we then have $\bar{J}_1 = J_{\bar{1}} = \{2, 2, 2\}$ and $\bar{J}_2 = J_{\bar{2}} = \{6, 6, 6, 7, 7, 7\}$.

Lemma 3 *For each sequence of orders in $\bar{c}$ with b internal breaks, first order's size s, and last order's size s', there is sequence of orders in c with b internal breaks, first order's size s, and last order's size s'.*

Proof Consider an arbitrary sequence $\bar{\pi}$ of orders in $\bar{c}$ with b internal breaks, first order's size s, and last order's size s'. Let $k_1, \ldots, k_{n_c^{\max}}$ be the positions of orders with the largest popular size $s_{\bar{c}}^*$ in $\bar{c}$. For each size s that occurs in c less than $n_c^{\max}$ times and, hence, is not represented in $\bar{c}$ and $\bar{\pi}$, we can insert an order $j \in J_c \setminus J_{\bar{c}}$ with $s_j = s$ in between two consecutive occurrences of s^*. Inserting these orders such that their sizes are strictly decreasing, we do not increase the number of internal breaks. Because there are $n_c^{\max} - 1$ gaps between consecutive occurrences of s^* and each size of orders in $J_c \setminus J_{\bar{c}}$ occurs at most $n_c^{\max} - 1$ times, those gaps suffice to host all orders in $J_c \setminus J_{\bar{c}}$. □

Lemma 3 states that each sequence of orders in reduced cart $\bar{c}$ can be interpreted as a sequence of orders in cart c with the same first and last orders' sizes and the same number of internal breaks. It can, therefore, adequately represent such sequences of orders in cart c, because they have the same contribution to the objective function by internal breaks and they have the same prerequisites for external breaks.

Example 3 (cont.): We consider sequences $\bar{\pi}_1 = (2, \underline{2}, \underline{2})$ and $\bar{\pi}_2 = (7, 6, \underline{6}, \underline{7}, \underline{7}, 6)$ for reduced carts $\bar{1}$ and $\bar{2}$, where a size is underlined if there is an internal break between its immediate predecessor and itself. Because cart 1 has only one popular size, there is only one sequence. Recall that we identify orders by their size only). Sequence $\bar{\pi}_2$ for reduced cart $\bar{2}$, however, is not unique. Positions of orders with the largest popular size are 1, 2, and 3 in $\bar{\pi}_1$ and 1, 4, and 5 in $\bar{\pi}_2$. The sequences can be interpreted as $\pi_1 = (2, 1, \underline{3}, 2, \underline{3}, 2)$ and $\pi_2 = (7, 6, 5, 4, \underline{6}, \underline{7}, 5, \underline{7}, 6)$ by inserting non-popular sizes in between these positions without causing any additional internal breaks. Note that different orders (and sizes) give rise to a break in $\bar{\pi}_1$ and π_1, but the total number does not change. Contrarily, the same orders (and sizes) give rise to a break in $\bar{\pi}_2$ and π_2, because there are no larger sizes than any popular size to be inserted.

It remains to be seen, however, if it suffices to consider reduced carts because all relevant sequences of orders in cart c are represented by sequences of orders in reduced cart $\bar{c}$. In the next step, we identify a set of dominant sequences of orders in c, which we will later show to be represented by sequences of orders in reduced cart $\bar{c}$.

Lemma 4 *For each sequence of orders in c with b internal breaks, first order's size s, and last order's size s', there is a sequence of orders in c*

- *with at most b internal breaks, first order's size $s'' \leq s$, and last order's size $s''' \geq s'$,*
- *with at most $b-1$ internal breaks, first order's size $s'' \leq s$, and last order's size $s''' < s'$,*
- *with at most $b-1$ internal breaks, first order's size $s'' > s$, and last order's size $s''' \geq s'$, or*
- *with at most $b-2$ internal breaks, first order's size $s'' > s$, and last order's size $s''' < s'$,*

where s'' and s''' are popular.

Proof Consider an arbitrary sequence π of orders in c with b internal breaks, first order's size s, and last order's size s'. Let s'' and s''' be the first and last popular size occurring in π, respectively. Note that s and s'' as well as s' and s''' may coincide. We proceed in two steps in order to derive a sequence π' with first order's size s'' and last order's size s'''. First, we delete orders in $J_c \setminus J_{\bar{c}}$ from the sequence, which leaves only orders in $J_{\bar{c}}$ in the same order as in π. Note that this step reduces the number of internal breaks by at least one, if $s'' \leq s$ and $s''' < s'$, it reduces the number of internal breaks by at least one, if $s'' > s$ and $s''' \geq s'$, and it reduces the number of internal breaks by at least two, if $s'' > s$ and $s''' < s'$. Second, we insert orders in $J_c \setminus J_{\bar{c}}$ into gaps between orders with the largest popular size $s^*_{\bar{c}}$, using the same procedure as in the proof of Lemma 3. We obtain sequence π' of orders in c with first order size s'', last order size s''', where s'' and s''' are popular. Since the procedure for reinserting orders in $J_c \setminus J_{\bar{c}}$ does not increase the number of internal breaks, as is shown in the proof of Lemma 3, sequence π' has one of the alternative properties listed in the lemma. □

Lemma 4 states that there is a set of dominant sequences with popular sizes as first and last orders' size. Sequence π' is dominant in the following sense: Starting with a larger order's size or ending with a smaller order's size bears the risk of an additional external break before or after the cart (as compared to π). However, for each potentially added external break π' has one internal break less.

Example 3 (cont.): We consider sequences $\pi_1 = (3, 1, \underline{2}, \underline{2}, \underline{3}, 2)$ and $\pi_2 = (5, \underline{7}, 6, \underline{6}, \underline{7}, 5, \underline{7}, 6, 4)$ for carts 1 and 2. Sequence π_1 starts with a non-popular size and ends with a popular size, while π_2 does neither start nor end with a popular size. Removing orders in $J_1 \setminus J_{\bar{1}}$ and $J_2 \setminus J_{\bar{2}}$, respectively, yields $\bar{\pi}_1 = (2, \underline{2}, \underline{2})$ and $\bar{\pi}_2 = (7, 6, \underline{6}, \underline{7}, \underline{7}, 6)$. Note that the number of internal breaks is reduced by one in both cases. In the case of π_2 and $\bar{\pi}_2$, this is due to the fact that $\bar{\pi}_2$ starts with a larger size than π_2. As seen in the example for Lemma 3, we obtain sequences $\pi'_1 = (2, 1, \underline{3}, 2, \underline{3}, 2)$ and $\pi'_2 = (7, 6, 5, 4, \underline{6}, \underline{7}, 5, \underline{7}, 6)$, which have one of the alternative properties (i.e., π'_1 has the first one and π'_2 has the third one).

Finally, we show that all these dominant sequences of orders in c are represented by sequences of orders in $\bar{c}$

Lemma 5 *For each sequence of orders in c with b internal breaks, first order's size s, and last order's size s', where s and s' are popular, there is a sequence of orders in $\bar{c}$ with at most b internal breaks, first order's size s, and last order's size s'.*

Proof Consider an arbitrary sequence π of orders in c with b internal breaks, first order's size s, and last order's size s', where s and s' are popular. We derive a sequence $\bar{\pi}$ of orders in $\bar{c}$ with at most b internal breaks, first order's size s, and last order's size s' by simply removing all orders in $J_c \setminus J_{\bar{c}}$. This procedure does not increase the number of internal breaks. □

Example 3 (cont.): Starting from $\pi'_1 = (2, 1, \underline{3}, 2, \underline{3}, 2)$ and $\pi'_2 = (7, 6, 5, 4, \underline{6}, \underline{7}, 5, \underline{7}, 6)$, as obtained in the example for Lemma 4, we derive $\bar{\pi}_1 = (2, \underline{2}, \underline{2})$ and $\bar{\pi}_2 = (7, 6, \underline{6}, \underline{7}, \underline{7}, 6)$.

Lemma 5 establishes that each sequence of orders in c in the dominant set according to Lemma 4 is represented by a sequence of orders in $\bar{c}$. Note that when interpreting sequence π', as obtained in the proof of Lemma 5, using the procedure in the proof of Lemma 3, we do not necessarily obtain π, but a sequence π'', which is equal with respect to relevant features (i.e., its number of internal breaks, first order's size, and last order's size).

We conclude Lemma 3, Lemma 4, and Lemma 5 as follows. It suffices to consider reduced cart $\bar{c}$ in any solution approach, because a dominant subset of sequences of orders in c is represented by sequences of orders in $\bar{c}$. After determining a solution for reduced carts, we can interpret it as a solution for original carts with the same objective value.

From this point on, we will consider reduced carts instead of carts. We will now derive that for reduced carts, too, only a subset of order sequences need to be considered. It is not hard to see that the only sequence $\pi_{\Delta_{\bar{c}}}$ for $\bar{c}$ achieving a minimum number of internal breaks is the one consisting of $n_c^{\max}$ subsequences, with each size occurring exactly once in each subsequence and orders sorted in decreasing sizes. It achieves $\Delta_{\bar{c}} = \Delta_c = n_c^{\max} - 1$ internal breaks.

Due to Lemma 2 it suffices to consider, in addition to $\pi_{\Delta_{\bar{c}}}$, sequences implying $\Delta_{\bar{c}} + 1 = n_c^{\max}$ internal breaks. They are considered next. Let s^+ be the next larger popular size to s, if $s \neq s^*_{\bar{c}}$.

Lemma 6 *Each sequence of orders in $\bar{c}$ with at most $\Delta_{\bar{c}} + 1$ internal breaks and first order's size $s \neq s^*_{\bar{c}}$ has a last order's size of at most s^+.*

Proof A sequence not starting with an order with the largest popular size cannot achieve $\Delta_{\bar{c}}$ internal breaks, because $\pi_{\Delta_{\bar{c}}}$ is the only sequence doing so and it starts with the largest popular size. Hence, each occurrence of the largest popular

size gives rise to an internal break and no other order does so. Thus, each size occurs in gaps between occurrences of the largest popular size exactly once, and only sizes smaller s occur after the initial order and before the first occurrence of the largest popular size. Hence, sizes larger s and smaller the largest popular size occur after the last occurrence of the largest popular size. □

We observe that for each reduced cart $\bar{c}$ and each of its popular sizes s, which is not the largest popular size, we can actually achieve a sequence of orders with exactly $\Delta_{\bar{c}} + 1$ internal breaks, first order's size s, and last order's size s^+. We construct such a sequence modifying $\pi_{\Delta_{\bar{c}}}$ by taking the first S' orders and move them to the end keeping their relative order. Here, S' is a number smaller than the number of popular sizes. We refer to the sequence obtained by having size s first as $\pi_{\Delta_{\bar{c}}}(s)$. If $s = s^*_{\bar{c}}$, then $\pi_{\Delta_{\bar{c}}}(s) = \pi_{\Delta_{\bar{c}}}$. Note that $\pi_{\Delta_{\bar{c}}}(s) = \pi_{\Delta_{\bar{c}}}$ dominates all other sequences starting with size s, because for give size s it achieves the minimum number of internal breaks and among those sequences it ends with the maximum order's size (minimizing potential for a following external break). We denote by

$$\Pi_c = \left\{\pi_{\Delta_{\bar{c}}}(s) \mid s \text{ is a popular size in } c\right\}$$

the set of dominant sequences for reduced cart $\bar{c}$.
Example 3 (cont.): For carts 1 and 2, we obtain

$$\Pi_1 = \left\{(2, \underline{2}, \underline{2})\right\} \text{ and}$$
$$\Pi_2 = \left\{(7, 6, \underline{7}, 6, \underline{7}, 6), (6, \underline{7}, 6, \underline{7}, 6, \underline{7})\right\}.$$

We conclude Lemma 2 to Lemma 6 in Theorem 3.

Theorem 3 *There is an optimum solution to each instance of PSMP, where the sequence of each cart c corresponds to a sequence in Π_c.*

Our solution methods will determine explicitly only sequences for reduced carts in addition to the sequence of carts themselves. Theorem 3 implies that we can restrict ourselves to sequences in Π_c for each cart c. In particular, this means that there is only one sequence to be considered, if c has only one popular size. Furthermore, we can see that there is tradeoff between first order's size and last order's size in Π_c. For sequences in Π_c with $\Delta_c + 1$ internal breaks, the first order' size is lower (which reduces potential for a preceding external break), if and only if last order' size is lower (which increases potential for a succeeding external break). Sequence $\pi_{\Delta_{\bar{c}}}$ maximizes the potential for both, a preceding external break and a succeeding external break, but simultaneously has one internal break less. We, thus, cannot identify any pairwise dominances between sequences in Π_c without further information. However, depending on the last order's size in the preceding cart, we can see that we can restrict ourselves to at most two sequences for each cart c as the following theorem states.

Theorem 4 *There is an optimum solution to each instance of PSMP, where for each cart c (but the first one) sequence $\pi_{\Delta_{\bar{c}}}$ or sequence $\pi_{\Delta_{\bar{c}}}(s)$ is chosen with s being the largest popular size of c smaller than the last order's size in the preceding cart.*

Proof Consider an optimum solution σ and the first cart c with neither $\pi_{\Delta_{\bar{c}}}$ or sequence $\pi_{\Delta_{\bar{c}}}(s)$ as sequence of orders. If there is an external break preceding c, we can replace its order sequence by $\pi_{\Delta_{\bar{c}}}$. We reduce the number of internal breaks by one, which compensates for a potential new external break succeeding c. If there is no external break preceding c, we can replace its order sequence by $\pi_{\Delta_{\bar{c}}}(s)$. We still have no external break preceding c, the number of internal breaks does not increase (because $\pi_{\Delta_{\bar{c}}}$ was not chosen for c), and the last order's size of c does not decrease (avoiding a new external break succeeding c). □

Theorem 5 *There exists an optimum solution where cart c precedes cart c', $c < c'$, in the cart sequence if c and c' coincide in their popular sizes.*

Proof Consider an optimum solution σ where c' precedes cart c. Note that potentially $\Pi_c \neq \Pi_{c'}$. Nevertheless, we can simply exchange c and c' if sequences for c and c' corresponding to $\pi_{\Delta_{\bar{c}}}(s)$ and $\pi_{\Delta_{\bar{c}'}}(s)$ for one popular size s have been chosen. This is due to the fact that both sequences coincide in their first sizes and in their last sizes. Exchanging c and c' keeping their respective sequences, hence, does not add any external breaks. Now assume that sequences for c and c' corresponding to $\pi_{\Delta_{\bar{c}}}(s)$ and $\pi_{\Delta_{\bar{c}'}}(s')$ for one popular sizes s and s' with $s \neq s'$ have been chosen. We, then, modify σ as follows. We choose sequences for c and c' corresponding to $\pi_{\Delta_{\bar{c}}}(s')$ and $\pi_{\Delta_{\bar{c}'}}(s)$ and switch c and c' in the cart sequence. No external break has been added since the sequence for c (c') after the modification starts and ends with same sizes as the sequence for c' (c) did before the modification. If sequences $\pi_{\Delta_{\bar{c}}}(s)$ and $\pi_{\Delta_{\bar{c}'}}(s)$ have the same number of internal breaks, then the number of internal breaks remained the same for both carts after the modification. If sequences $\pi_{\Delta_{\bar{c}}}(s)$ and $\pi_{\Delta_{\bar{c}'}}(s)$ have different numbers of internal breaks, then those numbers differ by one and $\pi_{\Delta_{\bar{c}}}(s)$ and $\pi_{\Delta_{\bar{c}'}}(s)$ have numbers of internal breaks differing by one, as well. The cart having one internal break less in σ has one internal break more in the modified solution, but the overall number of internal breaks remains the same. □

These structural properties of optimal solutions allow us to solve PSMP with a compact DP, which we elaborate in the following section.

5.2 An exact DP approach

Based on the groundwork of Sect. 5.1, we provide an efficient DP approach for solving PSMP to optimality. The basic idea of this DP is to attach carts one by one at the end of a sequence of already added carts. We propose to employ states $(\mathcal{C}, s)$ with

- $\mathcal{C} \subseteq \{1, \ldots, C\}$ representing the carts already attached and
- $s \in \{1, \ldots, S\}$ representing the last order's size of the last cart in the cart sequence.

Recall that we restrict ourselves to reduced carts due to Theorem 3. Hence, we have $\mathcal{O}(2^C \cdot \bar{S})$ states, where $\bar{S}$ is the number of sizes that are popular in any cart. The minimum number of (internal and external) breaks corresponding to state $(\mathcal{C}, s)$ is represented by $f(\mathcal{C}, s)$. The initial state is $(\emptyset, \cdot)$, with $f(\emptyset, S+1) = 0$ and dummy size $S+1$.
A transition reflects the attachment of a single cart not yet in $\mathcal{C}$ to the sequence. More specifically, a transition from state $(\mathcal{C}, s)$ to state $(\mathcal{C} \cup \{c\}, s')$ for order $c \notin \mathcal{C}$ and popular size s' of c represents adding $\bar{c}$ using sequence $\pi_{\Delta_{\bar{c}}}(\hat{s}) \in \Pi_c$ with $\hat{s}^+ = s'$, that is the sequence in Π_c that ends with size s'. Recall that restricting ourselves to sequences in Π_c is justified by Theorem 3. Due to Theorem 4, we consider a transition from state $(\mathcal{C}, s)$ to state $(\mathcal{C} \cup \{c\}, s')$ only if $\hat{s}$ is the largest popular size in c or if $\hat{s}$ is the largest popular size in c smaller than the last order's size s in the preceding cart.
Cost $c(\mathcal{C}, s, c, s')$ associated with this transition amount to the number of internal breaks associated with $\pi_{\Delta_{\bar{c}}}(\hat{s})$, increased by one if $\hat{s} \geq s$, that is if there is an external break before c.
Finally, we formulate the Bellman function as

$$f(\mathcal{C}, s) = \min \big\{ f(\mathcal{C} \setminus \{c\}, s') + c(\mathcal{C} \setminus \{c\}, s', c, s) \mid \text{the transition from } (\mathcal{C} \setminus \{c\}, s') \text{ to } (\mathcal{C}, s) \text{ is considered} \big\}.$$

Solving PSMP, then, corresponds to determining $\min\{f(\{1, \ldots, C\}, s) \mid s \text{ is a popular size}\}$. We have $\mathcal{O}(2^C \cdot \bar{S})$ states and $\mathcal{O}(2^C \cdot C \cdot \bar{S}^2)$ transitions and, thus, can determine the optimum solution in $\mathcal{O}(2^C \cdot C \cdot \bar{S}^2)$ time.
Example 1 (cont.): Fig. 7 depicts the DP graph for Example 1 introduced in Fig. 4a). Note that only states resulting from Theorem 2 are depicted. Furthermore, no states are created, in which cart C is scheduled before cart B, due to Theorem 5. The optimal solution depicted in Fig. 4c corresponds to the bold path from $(\emptyset, \cdot)$ to $(\{A, B, C, D\}, 1)$.
To further speed the DP up, we extend it to a bounded DP by integrating a state-specific lower bound and beam search heuristic for determining a quick upper bound (e.g., see Fliedner et al. 2011). For each state $(\mathcal{C}, s)$, we determine a simple lower bound on the number of breaks, assuming that no external breaks occur. In this way, we obtain

$$LB_{(\mathcal{C},s)} = f(\mathcal{C}, s) + \sum_{\bar{c} \in C \setminus \mathcal{C}} \Delta_{\bar{c}}.$$

In our beam search approach, we consider only the B states having the smallest lower bounds at each stage of the DP graph (i.e., a stage consists of all DP states having the same number of carts already serviced) for further exploration. All other states are deleted. In this way, a first heuristic solution is obtained. Depending on the choice of steering parameter B, beam search consumes not much computational time. We execute it before running the DP in order to determine an upper bound and consequently do not further explore states with a lower bound greater or equal than the upper bound obtained by beam search.

6 Computational performance

This section examines the performance of our solution methods. First, we describe our test instances in Sect. 6.1. Then, we examine the computational performance of the DP approach for PSMP in Sect. 6.2.

6.1 Test instances

Our computational tests are performed on different types of order data. We use (i) generated data, mainly for performance testing, to have a controlled environment where all influencing parameters can be systematically explored. Furthermore, we use (ii) a real dataset from a Brazilian e-commerce retailer, mainly to investigate managerial issues on representative order data.

(i) To derive *generated data*, we vary the following parameters that are the input to our data generator: cart capacity $|J_c| \in \{10, 20, 50, 100\}$, number of carts $C \in \{10, 15, 20\}$, and number of available carton sizes $S \in \{5, 10, 15\}$. For each parameter setting, we derive two types of instances, where the sizes are either uniformly distributed across all orders or follow a triangular distribution. For the former, we draw the size of each order uniformly distributed between 1 and S. For the latter, the size of each order is drawn from a triangular distribution with parameters $a = 1$, $b = S$, and $c = 0.1 \cdot S$. This distribution results in smaller sizes being selected more often, mimicking a warehouse scenario where smaller products (e.g., paperbacks) are ordered more often than larger ones (e.g., coffee table books). For each combination of parameters, we generate 375 instances, so we

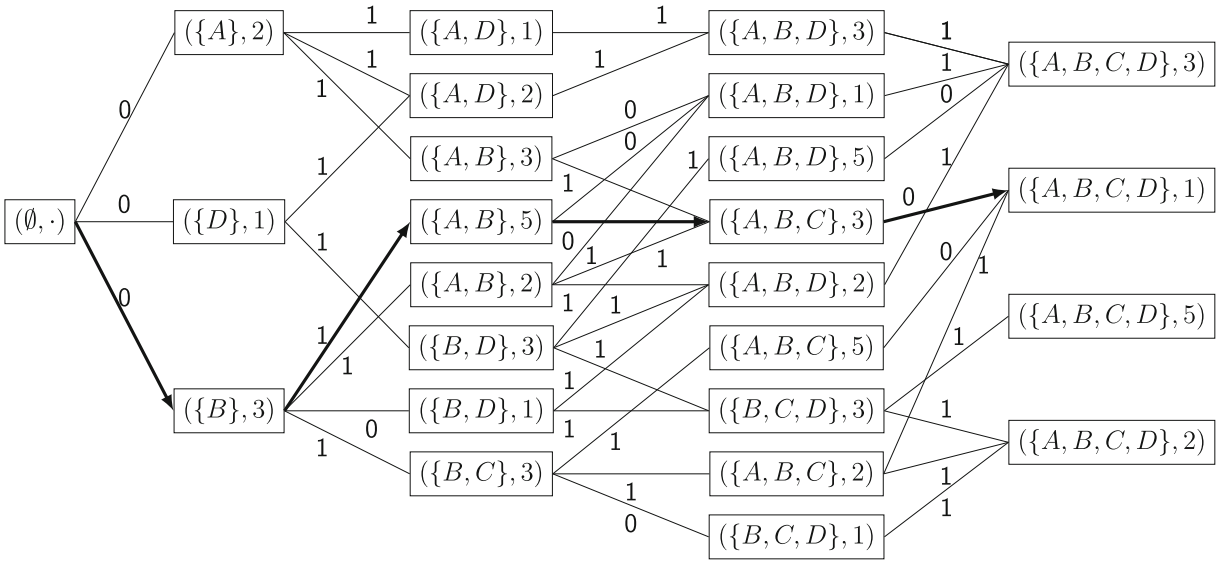


Fig. 7 DP graph for Example 1

have a total of $4 \cdot 3 \cdot 3 \cdot 2 \cdot 375 = 27,000$ instances in this dataset.

(ii) *Real-world data:* Further computational tests are based on real-world data from the e-commerce retailer Olist (2019). Their assortment for the years 2016 to 2018 consists of about 32, 000 products, whose detailed size information are publicly available. To generate 20 instances from this data, we do the following: For each instance, we randomly sample 300 products with a width between 8 and 40 cm. Each of these products represents a single-piece order, which we randomly partition among $C = 20$ carts, each with a capacity of $|J_c| = 15$ orders. Finally, the carton size of each product is determined by rounding up to the next available carton size of the instance.

All tests were performed on a PC running Ubuntu 22.04 with an Intel Core i7-12700K CPU with 5 GHz clock speed and 64 GB RAM. Preliminary tests, the results of which are not reported in this paper, have shown that choosing a beam width $B = 20 \cdot C$ provides a good compromise between quality and solution time for deriving the initial upper bound of DP. All algorithms and our data generator were coded in Java 18.

6.2 Performance results

This section reports on the computational performance of our solution methods based on the generated dataset.

First, we consider PSMP-fixed where the sequence of carts is given. Recall that this version of the problem can be solved in polynomial time by GREEDY (see Sect. 4). Even for the largest instances (i.e., with $|J_c| = 100$, $C = 20$, and $S = 15$, which should exceed the instance sizes of most warehouses using automated packing), the runtime is barely measurable.

Table 3 Runtimes of DP in CPU seconds

		S (triangular)			S (uniform)		
C	$\|J_c\|$	5	10	15	5	10	15
10	10	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
	20	<0.01	0.01	0.01	0.01	0.01	0.01
	50	<0.01	<0.01	0.01	0.01	0.01	0.01
	100	<0.01	<0.01	0.01	0.00	0.01	0.01
15	10	0.16	0.29	0.39	0.25	0.42	0.49
	20	0.12	0.24	0.32	0.23	0.36	0.44
	50	0.09	0.20	0.26	0.22	0.31	0.37
	100	0.07	0.17	0.22	0.20	0.27	0.31
20	10	9.68	19.19	27.13	15.53	28.97	36.20
	20	7.19	15.44	22.26	14.64	25.29	32.86
	50	5.05	12.13	17.38	13.70	22.60	28.44
	100	4.01	10.28	14.31	12.95	20.36	25.05

Thus, solving PSMP with a given cart sequence to optimality for even the largest warehouses is easily possible.

If determining the cart sequence is part of the problem, we apply the DP of Sect. 5.2 to solve PSMP to proven optimality. The runtimes, summarized in Table 3, suggest the following findings:

- *Highest impact of cart number C*: In line with our theoretical runtime analysis of DP, the solution time increases exponentially in C. However, since $C = 20$ carts (or bins, see Sect. 3.1) should definitely be at the top of what can be expected in real-world warehouses, runtimes well below one minute should not limit the use of DP in typical warehouses.

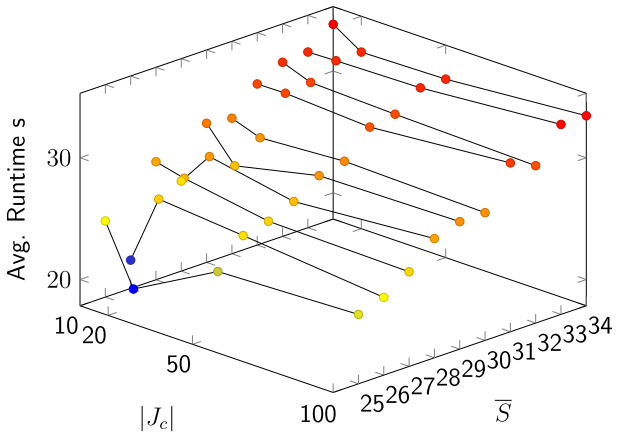


Fig. 8 Runtime of DP in CPU seconds depending on the number of popular sizes $\overline{S}$ and the number of jobs per cart $|J_c|$

- *High impact of the number of popular sizes* $\overline{S}$: The other influencing factor of the runtime of DP according to our theoretical analysis (see Sect. 5.2) is the number of popular sizes $\overline{S}$. Recall that only the (popular) sizes with the most orders per cart need to be considered. According to our computational results, their number increases, obviously, if we have more sizes S (i.e., more sizes increase the probability that more of them are popular), the size distribution is uniform (i.e., the triangular distribution produces a few highly demanded sizes, so that just a few become popular), and have small order capacity $|J_c|$ per cart. Especially, the later effect seems to counter intuition, because more orders (and thus larger instance sizes) seem to induce shorter runtimes.

The latter effect is further analyzed in Fig. 8. Here, the runtimes for instances with $C = 20$ carts and $S = 15$ carton sizes are depicted as a function of the jobs per cart $|J_c|$ and the total number of popular sizes $\overline{S}$ over all carts. Note that the lower bound on $\overline{S}$ is C, if each cart has exactly one popular size. In Fig. 8, we restrict ourselves to values of $\overline{S} = 25, \ldots, 34$, because values out of this range are rare. Instances resulting in the same number of popular sizes $\overline{S}$ are connected. In line with our theoretical runtime analysis of DP, we can observe that the runtimes remain (rather) stable for a constant number of popular sizes $\overline{S}$, irrespective of the number of jobs per cart $|J_c|$. As expected, the runtimes seem to increase almost linearly with an increase of popular sizes $\overline{S}$.

Therefore, we can conclude that the counterintuitive result of Table 3 (i.e., larger cart capacities $|J_c|$ decrease runtime) can thus be explained by the fact that more orders with their respective sizes tend to decrease the amount of popular carton sizes $\overline{S}$ of an instance. More random size selections for more orders on larger carts during data generation increases the probability that only a few of them will receive the highest demand and thus be popular. With only a few size selections, it is more likely that all sizes will have low demand and many of them will then be popular.

We conclude that, consistent with our theoretical analysis, the runtime of DP for solving PSMP with arbitrary cart sequence is mainly driven by the number of carts C and the number of popular sizes $\overline{S}$. However, even for warehouses where these numbers take their maximum value, the runtime of DP is small. Thus, DP seems well suited to solve even large PSMP instances of real-world size to proven optimality.

7 Managerial issues

This section is devoted to management issues. Specifically, we investigate the following three research tasks: (i) Once it has been decided how many different carton sizes should be available, the specific carton sizes need to be determined. To decide this important long-term issue, we use another DP approach to minimize the packaging waste for a given order set. Packaging waste arises by putting products into larger cartons than necessary to save on the number of employed carton sizes. With the DP, we can determine the packaging waste as a function of the number of available carton sizes. (ii) The complete waste–performance tradeoff of blocking machines is evaluated in Sect. 7.2. Here, we investigate to what extent more carton sizes reduce packaging waste but increase the performance loss (and vice versa). We quantify the performance loss by the blocking loss defined in Sect. 3, which counts the number of empty packing slots unused due to blocked packaging devices. (iii) Finally, Sect. 7.3 examines the value of sequencing flexibility. The two extremes are either no flexibility or full flexibility. In the former case, a fixed sequence of orders approaches the packaging machine without the possibility of changing it. Full flexibility (i.e., the given set of orders can be put into an arbitrary infeed sequence), on the other hand, promises the least performance loss, and inbound processes based on picking carts (with either a given or arbitrary cart sequence) lie somewhere in between the two extremes. We examine how switching to either of these inbound processes affects the throughput performance of a packaging machine.

7.1 What carton sizes to select?

Prior to operational scheduling, which is the main focus of this paper, long-term decisions must be made regarding the available carton sizes. The design of the blocking machine (and the resulting investment cost) is mainly influenced by the number of carton sizes to be provided, since each additional carton size requires another packaging device. We provide decision support on the right number of carton sizes in Sect. 7.2. There, we explore the waste–performance trade-

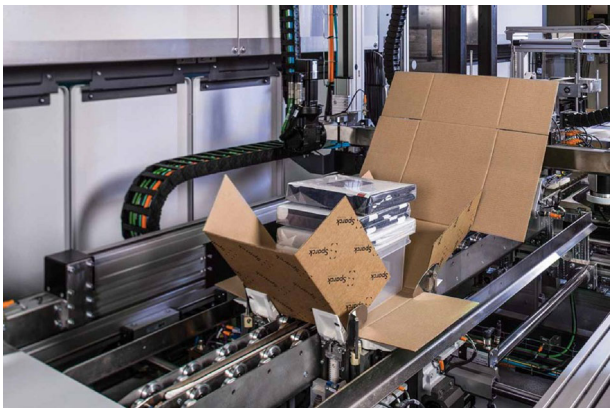


Fig. 9 Packaging machine during the carton folding process (Source: Sparck Technologies)

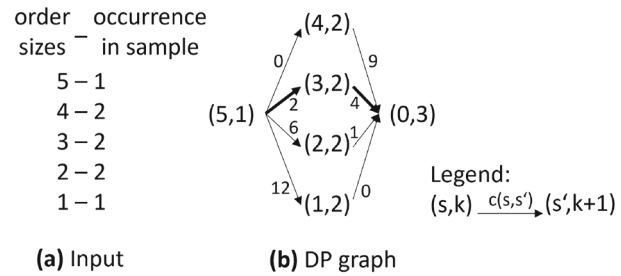


Fig. 10 Example for the DP solving the min-waste policy

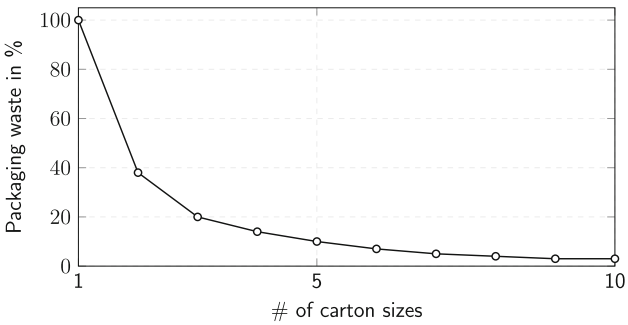


Fig. 11 Packaging waste per number of carton sizes for the real-world dataset

off that is mainly driven by the number of available carton sizes.

This section is dedicated to the choice of specific carton sizes, once the decision on the number of carton sizes to provide has already been made. If—based, for instance, on historical data—reliable information on the sizes of the orders to be packed and the frequency, in which each order size occurs, is available, then we can optimize the specific carton sizes, such that the total packaging waste gets minimized. Packaging machines fold the carton over the goods to be packed (see Fig. 9) and cutting of the carton at a suitable position ensures that in flow direction no packaging waste occurs. Across the flow direction, however, packaging waste arises whenever a good is packed into a carton of excessive width. This reduces the min-waste problem to a one-dimensional problem that only decides on the width of the cartons to be provided. Once the orientation of the orders to be packed is given, the resulting optimization problem can easily be solved by a straightforward DP. Recall that a similar DP has already been introduced by Lee et al. (2015) for optimizing the height of crates in the chemical industry (see Sect. 2). The DP proceeds as follows.

We employ states (s, k), which define that carton size $s \in \{1, \ldots, S\}$ is selected as the k-th carton size to be provided. For a given number of total carton sizes K to select, we thus have $\mathcal{O}(|S| \cdot K)$ states. We consider potential carton sizes from S to 1. As a result, we have a single starting state $(S, 1)$ and a single dummy end state $(0, K + 1)$. A transition from state (s, k) to (s', k'), with $s' < s$ and $k' = k+1$, reflects that all order sizes having order size $s \geq s_j > s'$ get assigned to carton size s. For each of those orders, a packaging waste of $s - s_j$ occurs. As a result, the cost $c(s, s')$, associated with the transition, equal the sum of the resulting waste.

Finally, we formulate the Bellman function as $f(s, k) = \min \left\{ f(s', k-1) + c(s', s) \mid s' > s \right\}$. Clearly, this DP determines the minimum packaging waste for the given order sizes in polynomial time.

Example 4 Based on the order data given in Example 1, Fig. 10a presents the resulting input data for the min-waste policy. The resulting DP graph, if $K = 2$ carton sizes are to be provided, is given in Fig. 10b. The bold-marked optimal solution leads to two carton sizes of size 5 and 3, with a total waste of 6, which equals the situation in Example 2.

Given this optimization approach, we can quantify the packaging waste for our real-world dataset (see Sect. 6.1) depending on the number of carton sizes that are provided. The results are plotted in Fig. 11. The performance metric reported here denotes the minimum waste determined by DP for the respective number of available carton sizes in relation to the waste if only a single one-size-fits-all carton is applied in %. The run times for applying the min-waste DP are near zero for every tested value of K, while it takes between 1.86s ($K = 1$) and 16.79s ($K = 10$) on average to apply the min-waste DP followed by solving the resulting instance by DP. These results lead us to our first managerial takeaway.

Finding 1: More carton sizes can significantly reduce the amount of packaging waste. However, the positive impact of additional carton sizes quickly diminishes. Adding a second carton size more than halves the amount of packaging waste, but adding an additional carton size when more than a handful are already available hardly contributes to any further significant reduction. This is good news for packaging machine users: There is no need to have an excessive num-

ber of carton sizes. With just a handful of sizes, most of the possible waste reduction can be achieved.

7.2 The waste–performance tradeoff

This section completes the picture on the waste–performance tradeoff and also includes the performance impact of the number of available carton sizes. Recall that more carton sizes tend to make it harder to properly sort the inbound orders according to demanded carton sizes on the infeed conveyor. Thus, more carton sizes not only promise a reduction of packaging waste—as the previous section has shown—but also tend to increase the blocking loss, where capacity is wasted and packaging slots must remain empty. More carton sizes come along with more back-to-back packaging devices that need to be installed along the infeed conveyor. Thus, more available carton sizes induce higher investment cost for the blocking machine and an increase of packaging capacity per packaging batch. To evaluate these expected coherences in our real-world data (i.e., solved with DP for the optimized carton sizes), Fig. 12 relates the number of available carton sizes to the performance side of the waste–performance tradeoff.

Specifically, these results relate to the following three basic performance metrics and how they are impacted by the number of available carton sizes, namely: (a) the makespan (i.e., defined by the number of packaging batches that are required to pack all orders), (b) the total blocking loss (i.e., defined by the total packaging capacity, obtained by multiplying the batches with the number of packaging devices, minus the number of orders), and (c) the blocking loss per packaging device (i.e., total blocking loss divided by the number of packaging devices). These results suggest the following findings:

(a) *Makespan:* More carton sizes to choose from requires more back-to-back packing devices, each dedicated to a specific size. More devices increase the packaging capacity, so we can validate the expected effect that the makespan for processing the orders of the real-world dataset decreases with more carton sizes (see Fig. 12a). However, their positive effect is diminishing, which is an indicator of the increasing difficulty of actually using the additional packaging capacity.
(b) *Total blocking loss:* The increase in total blocking loss the more carton sizes are available is visualized in Fig. 12b. Obviously, more and more packaging slots must remain empty because even our exact DP algorithm cannot properly sort the infeed sequence by carton size.
(c) *Blocking loss per packaging device:* Each additional packaging device added by another carton size causes an additional amount of blocking loss per device, as shown in Fig. 12c. We can see that the first additional carton sizes in particular lead to a significant increase in this performance metric. However, once a certain number of carton sizes are already available, each additional packaging device tends to add a fluctuating amount of blocking loss per device without a clear trend.

By relating these performance metrics to the packaging waste examined in the previous section, the full waste–performance tradeoff can be visualized using Fig. 13. Here, we show the efficient frontier for different numbers of available carton sizes in terms of both dimensions (i.e., waste and performance). Of course, the final choice of an appropriate number of carton sizes depends not only on these two dimensions, but also on the total cost of ownership of any additional packaging device. Because cost information varies widely among packaging machine manufacturers and can change over time, we will not include cost in our analysis. Instead, we end this section with the following take-home message:

Finding 2: More carton sizes reduce packaging waste, but also lead to more blocking loss, where more and more packaging slots must remain empty because the back-to-back packaging devices cannot receive orders properly sequenced by size. However, both effects diminish quickly, so introducing just a few (i.e., a handful in our data) carton sizes can provide a good compromise in the waste–performance tradeoff. This is good news for e-commerce retailers because it helps keep the necessary investment in back-to-back packaging equipment at a manageable level.

7.3 The impact of sequencing flexibility

Packing is just one stage of the complete order fulfillment process of warehouses and distribution centers. Before an order can be packed into a suitable carton, the ordered products must be retrieved either by a picker-to-parts or a parts-to-picker process (see De Koster et al. 2007, Lee et al. 2019). Depending on how these previous stages are connected to the infeed conveyor of a packaging machine, our PSMP faces different levels of sequencing flexibility (see also Sect. 3.1). Basically, there are the following alternatives:

(i) *Fixed infeed sequence:* No sequencing flexibility for our PSMP is available, if the preceding stages are directly connected by a conveyor that is fixedly attached to the infeed of the packaging machine. In this case, the infeed sequence equals the processing sequence of the previous stages, which is typically not optimized according to the needs of the packaging stage. We emulate this case by determining ten random order sequences for each instance of our real-world dataset, which cannot be altered by our PSMP. Their results are averaged.
(ii) *Given cart sequence:* The order transport between the preceding stages and packaging can also be organized in

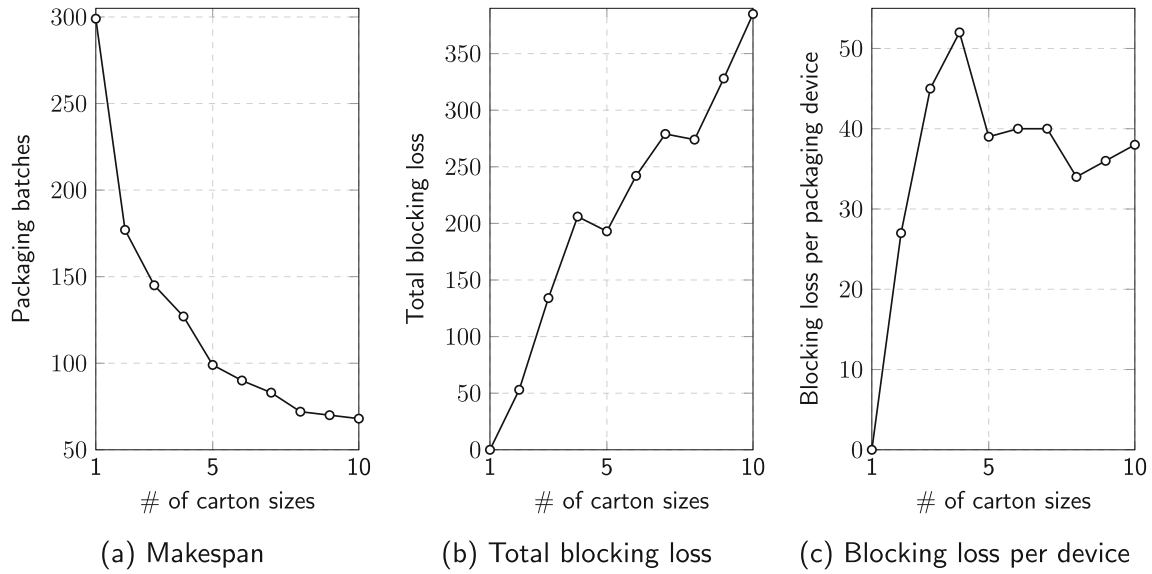


Fig. 12 Performance impact of different number of available carton sizes

a batchwise manner, e.g., via picking carts or bins. If the arrival sequence of these transport batches is given, the PSMP with given cart sequence is to be solved, and there remains the flexibility to optimize the infeed sequence of orders per cart. The add sequencing flexibility promises less blocking loss compared to (i) but comes at the price of double handling. The orders must be retrieved in a specific sequence from their batches to properly place them on the infeed conveyor. This produces extra search effort and manual handling. We emulate this case by drawing ten random cart sequences per instance, solving each of the resulting PSMP-fixed instances with GREEDY to optimality, and averaging the results.

(iii) *Arbitrary cart sequence:* Even more flexibility is offered, if also the sequences in which the batches (carts) are processed is part of the optimization by solving PSMP with arbitrary cart sequence. On top of the double handling, this also increases the demand for shop floor space for the intermediate storage of the batches. We emulate this case by solving PSMP with arbitrary cart sequence for each real-world instance.

(iv) *Perfect infeed sequence:* Finally, blocking loss can be avoided to the largest possible extent, if a random access on the complete order set is possible at the infeed station of the packaging machine. This either requires additional hardware (e.g., an automated storage and retrieval system (ASRS)) or excessive manual labor to retrieve the orders in arbitrary sequence from various batches. We emulate this case, by adding all orders to a single cart and solving the resulting instance with GREEDY (see Sect. 4).

Since it seems almost impossible to quantify the cost of these alternatives in an objective and generalizable way, we focus

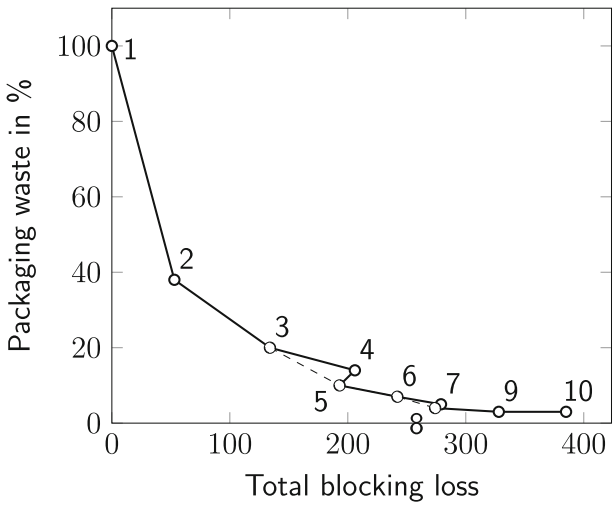


Fig. 13 Efficient frontier of different numbers of available carton sizes and their impact on packing waste and blocking loss

only on the performance impact of additional sequencing flexibility. In Fig. 14, we relate the makespan of alternatives (i), (ii), and (iii) to that of alternative (iv) and denote this performance metric 'increase in makespan in %'. The run times for (i), (ii), (iii) and (iv) are 0.17s, 0.28s, 17.6s and 0.6s on average. The average results for our real-world dataset show that a fixed infeed sequence that does not account for blocking loss of the packaging machine wastes a lot of packaging capacity and leads to an increase in makespan of more than 200% compared to the perfect infeed sequence. The performance loss of a batch transport to the packaging stage is much smaller, which leads us to the final managerial takeaway of this paper.

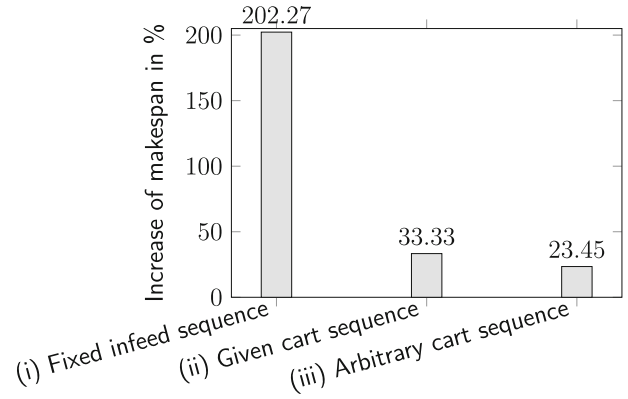


Fig. 14 Performance impact of sequencing flexibility

Finding 3: Batch transport of orders to the packing stage promises a good compromise between the higher investment cost of more sophisticated solutions based on random order access on all orders and the excessive blocking loss of given infeed sequences. Especially if the transport batches are not too small, this is still true if the batches are processed on a first-come, first-served basis based on a given cart sequence.

8 Conclusions

This paper focuses on the scheduling of e-commerce packaging machines. Specifically, we are the first to address the peculiarities of blocking machines, where multiple back-to-back packaging devices provide access to packaging cartons of different sizes. For various types of inflows, this paper defines the resulting order scheduling problem such that blocking loss (i.e., wasted packaging slots that cannot be utilized because the orders to be packed are not properly ordered according to carton sizes on the infeed conveyor) is minimized. We provide an in-depth analysis of the computational complexity and provide exact solution methods that provide optimal solutions in a runtime polynomial in the number of orders. Our performance analysis shows that these algorithms can solve even largest instances of real-world size to proven optimality in less than a minute. In addition, we investigate management issues and summarize the results into three main management takeaways.

From a theoretical perspective, future research should address the open case and resolve the complexity status of PSMP with arbitrary cart sequence when the number of carton sizes is part of the input. From a practical perspective, future research should systematically compare blocking machines with other types of packaging machines (e.g., setup machines) in terms of the waste–performance tradeoff. The latter, in particular, could make a valuable contribution to successfully reducing the environmental burden of excessive packaging waste in e-commerce.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Adler, L., Fraiman, N., Kobacker, E., Pinedo, M., Plotnicoff, J. C., & Wu, T. P. (1993). BPSS: A scheduling support system for the packaging industry. *Operations Research, 41*(4), 641–648.

Azadeh, K., De Koster, R., & Roy, D. (2019). Robotized and automated warehouse systems: Review and recent developments. *Transportation Science, 53*(4), 917–945.

Boysen, N., & de Koster, R. (2024). 50 years of warehousing research—An operations research perspective. *European Journal of Operational Research*. https://doi.org/10.1016/j.ejor.2024.03.026

Boysen, N., De Koster, R., & Weidinger, F. (2019). Warehousing in the e-commerce era: A survey. *European Journal of Operational Research, 277*(2), 396–411.

Boysen, N., & Stephan, K. (2016). A survey on single crane scheduling in automated storage/retrieval systems. *European Journal of Operational Research, 254*(3), 691–704.

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

Burkard, R. E., Deineko, V. G., Van Dal, R., van der Veen, J. A., & Woeginger, G. J. (1998). Well-solvable special cases of the traveling salesman problem: A survey. *SIAM Review, 40*(3), 496–546.

Coffman, E. G., Csirik, J., Galambos, G., Martello, S., & Vigo, D. (2013). Bin Packing Approximation Algorithms: Survey and Classification. In P. M. Pardalos, D.-Z. Du, & R. L. Graham (Eds.), *Handbook of Combinatorial Optimization* (pp. 455–531). Springer. https://doi.org/10.1007/978-1-4419-7997-1_35

De Koster, R., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research, 182*(2), 481–501.

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

Escursell, S., Llorach-Massana, P., & Roncero, M. B. (2021). Sustainability in e-commerce packaging: A review. *Journal of Cleaner Production, 280*, 124314.

Fliedner, M., Boysen, N., & Scholl, A. (2011). On the part inventory model sequencing problem: Complexity and beam search heuristic. *Journal of Scheduling, 14*, 17–25.

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

Janiak, A., Kovalyov, M. Y., & Portmann, M.-C. (2005). Single machine group scheduling with resource dependent setup and processing times. *European Journal of Operational Research, 162*(1), 112–121.

Lee, S. J., Chew, E. P., Lee, L. H., & Thio, J. (2015). A study on crate sizing problems. *International Journal of Production Research, 53*(11), 3341–3353.

Li, X., Gao, L., Pan, Q., Wan, L., & Chao, K.-M. (2018). An effective hybrid genetic algorithm and variable neighborhood search for integrated process planning and scheduling in a packaging machine workshop. *IEEE Transactions on Systems, Man, and Cybernetics: Systems, 49*(10), 1933–1945.

Li, S., Ng, C. T., & Yuan, J. (2011). Group scheduling and due date assignment on a single machine. *International Journal of Production Economics, 130*(2), 230–235.

Liu, Y., Wang, Z., & Cheng, G. Q. (2013). Optimization design for size of footwear outer packaging boxes. *Advanced Materials Research, 694*, 3516–3521.

Ng, C. T., Cheng, T. E., Janiak, A., & Kovalyov, M. Y. (2005). Group scheduling with controllable setup and processing times: Minimizing total weighted completion time. *Annals of Operations Research, 133*(1–4), 163–174.

Olist, (2019) . Brazilian e-commerce public dataset by olist. https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce (last access: January 2024).

Pfoser, S., Brandner, M., Herman, K., Steinbach, E., Brandtner, P., & Schauer, O. (2021). Sustainable transport packaging: Evaluation and feasibility for different use cases. *LOGI-Scientific Journal on Transport and Logistics, 12*(1), 159–170.

Singh, M., & Ardjmand, E. (2020). Carton set optimization in e-commerce warehouses: A case study. *Journal of Business Logistics, 41*(3), 222–235.