OR Spectrum (2013) 35:399–416
DOI 10.1007/s00291-012-0281-8

REGULAR ARTICLE

Positioning of loading units in a transshipment yard storage area

Florian Jaehn

Published online: 29 January 2012

Abstract This paper considers a decision problem as it arises in a rail–road transshipment yard when unloading a bundle of trains. The loading units, e.g. containers, arriving on trains and occasionally arriving on trucks have to be placed on storage lanes of limited capacity. Loading units are placed and removed keeping stacking and crane rail moves small. We present two NP-hard models and some heuristics for solving the problem. One of these heuristics is currently applied at the yard. The algorithms are then tested using real-life data.

Keywords Railway systems · Transshipment yard · Storage area · Heuristics

1 Introduction and literature review

We consider a problem as it arises at a large German transshipment yard. This yard is designed to support the intermodal transport as loading units can be moved from trains to trucks and vice versa. The typical layout of this and most other yards can be seen in Fig. 1.

Our yard consists of four parallel tracks for trains, two lanes for trucks and two storage lanes for an intermediate storage of units. Two rail mounted gantry cranes allow the movement of loading units (such as containers, swap bodies, or semi-trailers) from each lane/track to another one. Loading units are preferably moved directly from a truck to a train. However, trucks with goods designated to a certain train frequently

This work has been supported by the German Science Foundation (DFG) through the Grant "Optimierung der Containerabfertigung in Umschlagbahnhöfen" (BO 3148/1-1 and PE 514/16-1).

F. Jaehn (✉)
Institut für Wirtschaftsinformatik, Universität Siegen,
Hölderlinstraße 3, Siegen 57068, Germany
e-mail: florian.jaehn@uni-siegen.de

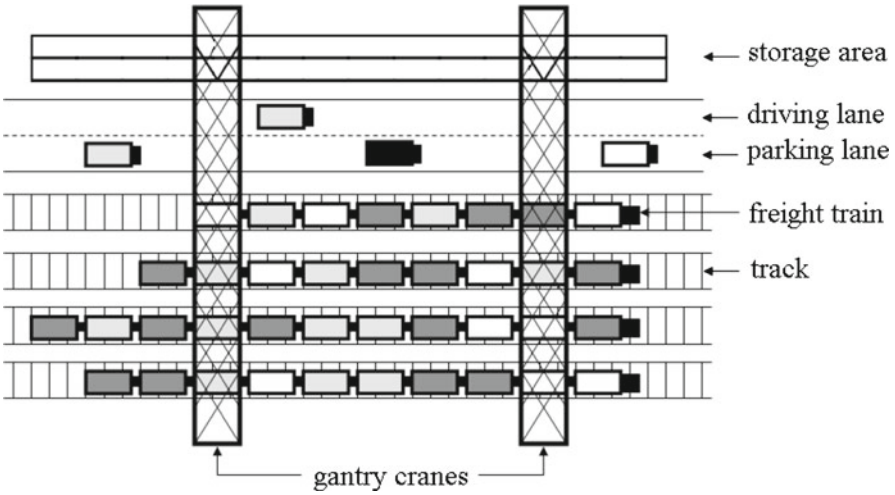

Fig. 1 Schematic representation of a rail–road transshipment yard

arrive much earlier than the receiving train. Thus, in most cases, the loading unit is placed in the storage area and a direct move from a truck to a train is rather exceptional. Similarly at the arrival of a train, most trucks which receive a loading unit from this train have not appeared. Therefore, the lion's share of all loading units is dropped at the storage area where they stay on average about 24 h. However, the variance of the dwell time is rather high.

A peculiarity of transshipment yards is that trains are processed in bundles (Bostel and Dejax 1998). This implies that trains entering the yard together will also leave the yard at the same time. During processing of these trains, no other trains may enter or leave.

Thus, processing of loading units at transshipment yards requires solutions to the following five decision problems, cf. Boysen and Fliedner (2010):

(i) Determine the bundles of trains and their sequence for entering the yard.
(ii) Assign each train to a railway track.
(iii) Decide on positions of the loading units on trains.
(iv) Assign moves of loading units to gantry cranes.
(v) Decide on the sequence of moves per crane.

As all of these problems are somehow interdependent, a holistic solution approach might be desirable. However, it is rather questionable whether a system covering all problems is capable of considering all interactions such that an optimization delivers good solutions quickly. Many of the above-mentioned problems are NP-hard by itself (e.g. Boysen et al. (2010) show NP-hardness of two problems of type (i)). Thus, it is common practice to use the list as a hierarchical decomposition of transshipment yard processing and solve each problem separately.

An analysis of our transshipment yard led to another decision problem of practical importance:

(vi) Decide on positions of the loading units in the storage area.

This paper is, to the best of our knowledge, the first one to treat problem (vi). Its solution has a direct impact on the efficiency of yards. The efficiency of the yard at hand is measured by the crane rate, i.e. the number of times a loading unit is moved by a crane, and by the operating times of the cranes.

The positioning of loading units in the storage area influences both measures. Firstly, a bad positioning of the loading units can increase the crane rate. If the storage area is too packed to stock a loading unit on the floor, it can be stacked one on top of the other. However, if the bottom unit is to leave the yard before the one on top, an extra move is necessary. Thus, as few loading units as possible should be set on top of other units.

Secondly, as the operating time of a crane additionally depends on the loading unit's position in the storage area, crane moves along the tracks are unproductive while cross traverse moves of cranes are inevitable. Moreover, a time-consuming maintenance of a crane has to be performed whenever the crane rail moves add up to a certain distance. An optimal storage management is a trade-off between loading unit stacking and transport to keep the cranes' operating time limited.

As mentioned before, literature on transshipment yards does not cover problem (vi). There are several results on problems (i)–(v). Boysen et al. (2010, 2011) treat problem (i) for a rail–rail transshipment yard. The assignment of trains to tracks (problem (ii)) is analyzed by Alicke and Arnold (1998). There are approaches deciding on the positions of loading units on the trains (see e.g. Corry and Kozan (2006), Corry and Kozan (2008), Bruns and Knust (2010)). Many of these approaches consider weight and length restrictions and/or optimize the aerodynamics of the running trains. Papers treating problem (iv) are presented by Kozan (1997) and Boysen and Fliedner (2010). Finally, problem (v) is analyzed by Montemanni et al. (2009). Furthermore, there are some models simultaneously solving more than one problem (e.g. Souffriau et al. (2009) treat problems (iv) and (v)).

Problem (vi) has somewhat been treated in literature on container handling at seaports. However, there are some differences. At seaports, containers are stacked in several layers in the storage area. Contrary, in the transshipment yard at hand, most loading units are placed on ground and stacking is rather exceptional and only requires two tiers. Thus, stacking rules (see e.g. Dekker et al. 2006; Borgman et al. 2010) and an extensive reshuffling when picking a loading unit from the storage area are not required (see e.g. Kim et al. (2000); Kim and Hong (2006); Caserta et al. (2009) for such procedures). Furthermore, at transshipment yards, the variety of different lengths of loading units is much higher. While many seaports handle only three different types of containers, loading units at our transshipment yard are classified into 18 lengths. Hence, storage problems arising at seaports cannot be transferred to rail transshipment yards. For a more detailed description on the operational processes at seaports, we refer to the surveys by Steenken et al. (2004) and Stahlbock and Voß (2008).

The remainder of this paper is organized as follows. Two formal models of the problem are presented in Sect. 2 followed by complexity proofs. In Sect. 3, we propose algorithms, which are tested on real-life data in a numerical study (Sect. 4). Finally, a summary and an outlook on future research appears in Sect. 5.

2 Problem description

We consider a transshipment yard as it is shown in Fig. 1. There are no restrictions on the number of tracks, but we assume that there are exactly two storage lanes. Note that this assumption is only for simplicity reasons and the models to be presented can easily be extended for different numbers of storage lanes. All storage lanes have the same length, which is denoted by L. A certain position on a storage lane can be described by a real number from the interval $[0, L]$.

Assume that a bundle of trains enters the yard. These trains transport a set of loading units C to be moved to the storage area. No loading units are put on the trains until all trains are unloaded. For each unit $i \in C$, its length p_i is known and its width is irrelevant because they are never turned and always positioned on the lane parallel to their position on the train. The center of unit i of length p_i can receive a lane position $M(i)$ in the interval $\left[\frac{p_i}{2}, L - \frac{p_i}{2}\right]$ different to the preferred position o_i, which is either the perpendicular projection of the unit's train position onto the storage lanes or a predefined position.

Usually, the storage area is not empty, and set C^{fix} may contain those loading units already placed in the storage area. For each unit $i \in C^{\text{fix}}$, we know its lane $l(i) \in \{1, 2\}$ where it is kept, its length p_i, the horizontal position of its center $M(i)$ ($\frac{p_i}{2} \leq M(i) \leq L - \frac{p_i}{2}$), and its left and right end on the lane $S(i) = M(i) - \frac{p_i}{2}$ and $E(i) = M(i) + \frac{p_i}{2}$, respectively.

To prevent that our model is filled with numerous real-life characteristics that might not have much influence on the decision, we make several simplifying assumptions.

1. We assume that decision problems (i)—(v) are solved and we consider their solutions as input data.
2. The objective of minimizing the crane rate is attained by minimizing the number of loading units piled up. To some degree, this is a simplification. Piled up loading units lead to a stacking operation but need not lead to an additional crane move if the top unit leaves the yard earlier than the bottom one. However, as there is a high variance in the dwell time of loading units, stacking is likely to cause an extra move.
3. The second objective, the minimization of operating times of a crane (per loading unit), is mostly influenced by other decisions to be made at the yard, especially by the sequence of crane moves. Note that at our yard cranes unload the trains from "left to right", which is due to several operational requirements. Therefore we may assume that the sequence of unloading units is given. For a given unloading sequence, the influence of positioning loading units in the storage area on the operating times may be roughly proportional to the distance cranes move (lengthwise) along the tracks on the crane rail. This is certainly a simplification as a situation may appear, in which a longer lengthwise move in the "right" direction might be better than a short one in the "wrong" direction. However, moving the gantry crane completely instead of only its grab leads to a higher maintenance frequency, which has a massive impact on the operating times. Thus, minimization of crane rail moves generally leads to a reduction of the operating times. This objective can easily be measured by $|o_i - M(i)|$, i.e. the distance between the units' original (lengthwise) position on the train and the designated position in the storage area.
4. We assume that a loading unit can be stacked anywhere on top of other units, i.e. its position in the second layer can be chosen arbitrarily. In practice, loading units in

the second layer must not jut out and must follow certain stability constraints. E.g a large container (2 TEU) must not be placed on top of a small container (1 TEU), but may be set on top of two smaller containers. It might not even be possible to place a small container on top of a larger container, because both become unstable. However, stacking is rather exceptional at the considered terminal and usually less than 15% of the units are located in the second tier. Thus, in practice it is usually not a problem to find a suitable spot on top of other units nearby, especially as loading units of different sizes may be stacked as long as the anchoring devices are in the same distance. Therefore, this is a less restrictive assumption.

5. There are two cranes at the yard that are not able to pass each other. Thus, moving a loading unit from one end of the yard to the opposite one causes a crane operating conflict. However, we do not consider any interferences between the two cranes in this problem setting. In practical situations, crane interferences can easily be avoided by assigning fixed areas to cranes. In this case, the problem at hand can be applied to each crane area separately. Another option is that one crane starts unloading at the very left end of the yard, the other one in the middle of the yard and then both move to the right, which is the common process at our yard. Each crane then exclusively processes on one half of the yard and the storage management of each half can be processed separately. Thus, in the following we will assume that there is only one crane.
6. We only treat the problem of placing loading units in the storage area coming from trains but not from trucks. As there are no restrictions on the size of set C, the question of placing loading units from trucks can be seen as a special case. Furthermore, this assumption is dropped later on.
7. For the time being, we consider a static model. This means that we do not take into account that the positioning of loading units in the storage area might influence later decisions, when new train bundles enter the yard. Because this assumption is of little practical relevance, we will drop it later on.

Based on these assumptions and using the notation as described in Table 1, we can present the static yard storage problem (SYSP):

$$\text{(SYSP)}\ \min \alpha_1 \sum_{i \in C \cup C^{\text{fix}}} \sum_{j \in C \cup C^{\text{fix}}} \frac{x_{ij}}{2} + \alpha_2 \sum_{i \in C} |o_i - M(i)| \tag{1}$$

subject to

$$(1 - x_{ij})(1 - |l(i) - l(j)|)\,(\Theta(E(i) - S(j)) + \Theta(E(j) - S(i))) \leq 1 \quad \forall i, j \in C \cup C^{\text{fix}}, i \neq j \tag{2}$$

$$E(i) \leq L \quad \forall i \in C \tag{3}$$

$$S(i) \geq 0 \quad \forall i \in C \tag{4}$$

$$l(i) \in \{1, 2\} \quad \forall i \in C \tag{5}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in C \cup C^{\text{fix}} \tag{6}$$

Table 1 Notation

C	Set of loading units to be placed in the storage area
C^{fix}	Set of loading units already placed in the storage area
p_i	Length of loading unit $i \in C \cup C^{\text{fix}}$
$p^{\max}$	Upper bound for the length of each loading unit, $p^{\max} := \max_{i \in C \cup C^{\text{fix}}} \{p_i\}$
L	Length of each of the two storage lanes
o_i	Predefined position of the center of loading unit $i \in C$
$l(i)$	The storage lane in which unit $i \in C \cup C^{\text{fix}}$ has been placed. $l(i) \in \{1, 2\}$. Decision variable for all $i \in C$
$M(i)$	Position of the center of unit $i \in C \cup C^{\text{fix}}$ with $\frac{p_i}{2} \leq M(i) \leq L - \frac{p_i}{2}$. Decision variable for all $i \in C$
$S(i)$	Position of the left end of unit $i \in C \cup C^{\text{fix}}$. $S(i) := M(i) - \frac{p_i}{2}$
$E(i)$	Position of the right end of unit $i \in C \cup C^{\text{fix}}$. $E(i) := M(i) + \frac{p_i}{2}$
x_{ij}	Binary variable: 1, if loading units i and j ($i \neq j$) are piled up on the same storage lane; 0, otherwise
α_1	Given weight for the objective of minimizing units piled up. $\alpha_1 \geq 0$
α_2	Given weight for the objective of minimizing crane rail moves. $\alpha_2 \geq 0$
$\Theta(a)$	Heaviside function: $\Theta(a) = 1$ if $a > 0$, $\Theta(a) = 0$ else

The objective function (1) counts the number of stacked loading units, described by binary variable x_{ij}, and the total distance of crane rail moves. The latter is simply the deviation of the loading unit's position in the storage area $M(i)$ from its projected position o_i on the train. We may assume that the objectives are sorted in lexicographic order. To make the model more general, the objectives are merged using a linear combination. By choosing $\alpha_1 \gg \alpha_2$, a lexicographic order can easily be obtained. Restriction (2) ensures that binary variable x_{ij} is either one, or loading units i and j are on different lanes in the storage area, or they do not overlap horizontally. Note that this restriction can easily be linearized. Restrictions (3) and (4) ensure that each loading unit does not exceed the limits of the storage area. There are two lanes in the storage area, which is enforced by restriction (5). Finally, it is enforced that x_{ij} is binary using restriction (6). We will refer to a solution of SYSP using the triple (M, l, x), containing the decision variables $M(i)$ and $l(i)$ for all $i \in C$ and variables x_{ij}, which are directly obtained from the decision variables.

Note that finding a feasible solution of SYSP is easy. In the model, stacking can be done arbitrarily high such that $M(i) := o_i$ and $l(i) = 1$ for all $i \in C$ is feasible (but undesirable). At least from a theoretical point of view, SYSP is hard to solve. We will now show that SYSP is NP-complete in the strong sense using a reduction of the well known 3-Partition problem, which is unary NP-hard (Garey and Johnson 1979).

Theorem 2.1 *SYSP is NP-complete in the strong sense.*

Proof For the proof, we need the 3-Partition problem:

3-Partition Problem. Consider integer numbers B, m, and a multiset of $3m$ integers $A_1, \ldots, A_{3m}$ such that $B/4 \leq A_i \leq B/2$ for all $i \in \{1, \ldots, 3m\}$ and $\sum_{i=1}^{3m} A_i = mB$. Is it possible to partition the multiset $\{A_1, \ldots, A_{3m}\}$ into m disjoint subsets such that the sum of the numbers in each subset equals B?

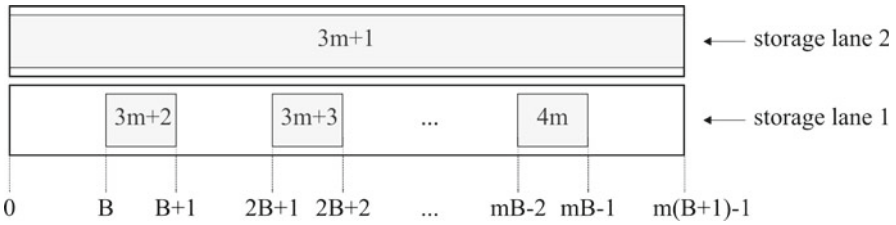

Fig. 2 According instance of SYSP for reducing 3-Partition

Obviously, SYSP is in NP. From a given instance of the 3-Partition problem, we will now construct an instance of SYSP as follows:

$\alpha_1 = 1, \alpha_2 = 0, L := m(B+1) - 1, C := \{1, \ldots, 3m\}$ and $p_i := A_i$ for all $i \in C, C^{\text{fix}} := \{3m+1, \ldots, 4m\}$. The first of the m elements of C^{fix} is to block a whole storage lane. Thus, $l(3m+1) = 2, S(3m+1) = 0$, and $p_{3m+1} = L$. The remaining elements are to block the space of the first storage lane in such a way that m slots of length B remain (see Fig. 2). This is obtained by setting $p_i = 1$ and $l(i) = 1$ for $3m+2 \leq i \leq 4m$, $S(3m+2) = B$, $S(3m+3) = 2B+1$, $S(3m+4) = 3B+2$, etc.

Obviously, there is a solution of SYSP in which $x_{ij} = 0$ for all $i, j \in C \cup C^{\text{fix}}$, if and only if the instance of 3-Partition has a solution. □

For the proof, it is inevitable that C^{fix} is not empty. If we assume $C^{\text{fix}} = \emptyset$ and if two storage lanes are considered, then SYSP is NP-complete in the ordinary sense. This is proved by reducing the Partition problem to SYSP. The Partition problem is binary NP-complete (Karp 1972).

Theorem 2.2 *The special case of SYSP in which $C^{\text{fix}} = \emptyset$ is NP-complete in the ordinary sense.*

Proof Partition problem. Given a multiset $A = \{A_1, \ldots, A_m\}$ of m integers. Is it possible to partition A into two disjoint subsets $A^{(1)}$ and $A^{(2)}$ such that $B := \sum_{A_i \in A^{(1)}} A_i = \sum_{A_i \in A^{(2)}} A_i$?

Given an instance of the partition problem, we construct an instance of SYSP:

$\alpha_1 = 1, \alpha_2 = 0, L := B, C := \{1, \ldots, m\}$ and $p_i := A_i$ for all $i \in C$. An assignment of an integer to one of the subsets then corresponds to an assignment of the corresponding loading unit to one of the storage lanes. Obviously, there is a solution of SYSP in which $x_{ij} = 0$ for all $i, j \in C$, if and only if the instance of Partition has a solution. □

Note that both proofs only use the objective of minimizing overlapping loading units. If the lengthwise moves were the only objective, i.e. $\alpha_1 = 0$ and $\alpha_2 = 1$, then the problem could easily be solved by setting $M(i) = o_i \; \forall i \in C$.

As mentioned before, the assumption of a static unloading situation is unrealistic. Thus, we will now present a model in which loading units are picked up and delivered by trucks during the unloading process of a bundle of trains. To obtain a model, which is applicable for practical situations, but is still not overloaded, the following, additional assumptions are made.

Table 2 Notation

T	Number of time periods or number of loading units to be unloaded from trains, $T = \|C\|$
C_t	Set of loading units to be placed in the storage area in time period $t \in \{1, \ldots, T\}$
C_t^{fix}	Set of loading units in the storage area at the start of time period $t \in \{1, \ldots, T\}$
Ξ_t	Random variable exposing a (possibly empty) set ξ_t of loading units from trucks to be placed in the storage area in time period $t \in \{1, \ldots, T\}$
H_t	Random variable exposing a (possibly empty) set η_t of loading units being picked up in time period $t \in \{1, \ldots, T\}$

1. For the dynamic model it is important to know the sequence in which the loading units are unloaded from the trains. Thus, C is no longer treated as an unordered set of loading units, but as a given sequence of loading units $C = \{c_1, c_2, \ldots, c_T\}$ with $T := |C|$. The assignment of the storage lane and the precise position has to be done successively according to the order of C. In fact, if C were ordered in the SYSP as well, the problem and its results would not change: For each loading unit of SYSP, it has to be decided on the horizontal position and on the storage lane. In SYSP, it makes no difference in which sequence these decisions are made and thus, we may assume a given one.
2. We consider T periods, in each of which one loading unit c_t $(1 \leq t \leq T)$ has to be unloaded. During one period, additional crane moves from or to a truck may be necessary (see below) so that not all periods need to be of same length. The set of units within the storage area at the start of period t is denoted by C_t^{fix}.
3. The planning horizon is T and we do not consider the time after T when trains are loaded and the storage area becomes less densely packed.
4. At the start of each time period t, the realization ξ_t of a random variable Ξ_t describes a (possibly empty) set of loading units that additionally have to be assigned to the storage area. We assume that the realization of Ξ_t is not known until the start of period t, but its distribution over all possible types of loading units is known. That means that we know the distribution of the number of loading units, the distribution of their lengths, and the distribution of the lengthwise point of delivery o_i. The units to be placed in the storage area during period t are $C_t := \{c_t\} \cup \xi_t$.
5. At the end of each time period t, i.e. after c_t and all units described by ξ_t have been moved to the storage area, some loading units need to be delivered from the storage area to trucks. This process is described by a random variable H_{t+1}. We may assume that for each realization η_{t+1}, $\eta_{t+1} \subseteq C_t^{\text{fix}} \cup C_t$ holds. Again, we assume that the distribution of each H_t $(2 \leq t \leq T)$ is known, but its realization η_t is unknown until the start of time period $t - 1$. As the positioning of rail-cars and trucks is outside of the scope of this paper, the distance of crane rail moves delivering loading units from η_t need not be added to the objective function.
6. We keep the second assumption of the static model, namely counting the number of piled up loading units instead of counting the precise number of extra crane moves.

Some additional notation is presented in Table 2, leading to the dynamic yard storage problem (DYSP):

$$\text{(DYSP)}\ \min \sum_{t=1}^{T} \left(\alpha_1 \sum_{i \in C_t \cup C_t^{\text{fix}}} \sum_{j \in C_t \cup C_t^{\text{fix}}} \frac{x_{ij}}{2} + \alpha_2 \sum_{i \in C_t} |o_i - M(i)| \right) \tag{7}$$

subject to

$$(1 - x_{ij})(1 - |l(i) - l(j)|)\,(\Theta(E(i) - S(j)) + \Theta(E(j) - S(i))) \leq 1$$
$$\forall i, j \in C_t \cup C_t^{\text{fix}}, \quad i \neq j, 1 \leq t \leq T \tag{8}$$
$$E(i) \leq L \qquad \forall i \in C_t \tag{9}$$
$$S(i) \geq 0 \qquad \forall i \in C_t \tag{10}$$
$$l(i) \in \{1, 2\} \qquad \forall i \in C_t \tag{11}$$
$$C_t = \{c_t\} \cup \xi_t \qquad 1 \leq t \leq T \tag{12}$$
$$C_1^{\text{fix}} = C^{\text{fix}} \tag{13}$$
$$C_t^{\text{fix}} = \left(C_{t-1}^{\text{fix}} \cup C_{t-1}\right) \setminus \eta_t \quad 2 \leq t \leq T \tag{14}$$
$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in C_t \cup C_t^{\text{fix}},\ 1 \leq t \leq T \tag{15}$$

The objective function (7) is to minimize the stacked loading units and the total distance of crane rail moves. However, contrary to SYSP, loading units arriving from trucks are considered as well. This is enforced by restrictions (12). Restrictions (13) and (14) define the fixed loading units in each period. The remaining restrictions (8)–(11) and (15) are equivalent to those of SYSP. Let (M_t, l_t, x_t) denote a solution of period t of an instance of DYSP.

If $P(\xi_t = \emptyset) = P(\eta_t = \emptyset) = 1$ for all $t \in \{1, \ldots, T\}$, i.e. no trucks deliver or pick up any loading units, then SYSP is a special case of the DYSP and therefore the DYSP is NP-hard.

Note that the DYSP is unary NP-hard even if $T = 1$. This is obvious as the set of loading units to be placed in the storage area, $\xi_1 \cup \{c_1\}$, can be any arbitrary set. Thus, SYSP is equivalent to DYSP with $T = 1$. That means solving the DYSP is equivalent to solving a "hard" problem in each period. Thus, each algorithm for solving DYSP can also be used for solving SYSP and vice versa.

3 Algorithms

In our transshipment yard, a rule-based approach is applied to position loading units in the storage area. However, it has been discussed whether it might be reasonable to use a grid pattern, which means that the storage area is split into small sections, each of which holds exactly one loading unit.

3.1 Rule-based approach

At the moment, the crane operator decides on the position where to drop off a loading unit in the storage area observing the only rule that a new loading unit must adjoin another one. Quite obviously the operator uses preferably the next available spot such that the unit adjoins another one. We will recall to this procedure as a rule- based approach, which can be formalized as follows for solving DYSP.

Algorithm 1 (Rule-based approach)

In every period do:

1. Initialization: Sort the set C_t according to arrival time.
2. Iteration: For each element $i \in C_t$ go to Step 3.
3. Positioning: Determine all combinations of $M(i)$ and $l(i)$, such that one of the following conditions holds.
- $S(i) = 0$
- $l(i) = l(j)$ and $S(i) = E(j)$ for some $j \in C_t^{\text{fix}}$
- $E(i) = L$
- $l(i) = l(j)$ and $E(i) = S(j)$ for some $j \in C_t^{\text{fix}}$

Among these solutions, determine those for which $\left\{ j \in C_t^{\text{fix}} \mid x_{ij} = 1 \right\}$ is minimized. If a solution is not unique, apply the one among these for which $|o_i - M(i)|$ is minimized. Go to Step 4.
4. Update: Set $C_t^{\text{fix}} = C_t^{\text{fix}} \cup \{i\}$.

Although this algorithm uses the available space efficiently when unloading a train to an empty storage area, it is rather myopic concerning loading units that are removed from the storage area. The resulting gaps might not be of adequate size for the incoming loading units. Furthermore, the gaps are filled rather greedily but not anticipatory.

3.2 Grid pattern

The idea of this algorithm is to sketch a grid on the ground of the storage area. Each section can capture exactly one loading unit. Sections need not have the same lengths, but the number of different lengths is naturally bounded by the number of different loading unit lengths. It might appear that a section is longer than the loading unit assigned to it. Thus, a storage lane cannot hold as many units as if the units adjoin each other, but with a high probability gaps can be filled with incoming units.

For a given grid pattern, the algorithm works as follows:

Algorithm 2 (Grid Pattern)

In every period do:

1. Initialization: Sort the set C_t according to arrival time.
2. Iteration: For each element $i \in C_t$ go to Step 3.

3. Positioning: For each section s being large enough to hold this loading unit, determine the objective functions values for the case that this loading unit is assigned to section s.
Among these solutions, determine a best one in which $\left\{ j \in C_t^{\text{fix}} \mid x_{ij} = 1 \right\}$ is minimized, the length of s is minimized, and finally in which $|o_i - M(i)|$ is minimized. Go to Step 4.

4. Update: Set $C_t^{\text{fix}} = C_t^{\text{fix}} \cup \{i\}$.

The algorithm only positions a loading unit in a section bigger than necessary, if it avoids stacking this unit. Among all free sections of the smallest feasible size, the nearest (concerning crane rail moves) is selected.

The determination of the lengths of the sections, the number of sections of each length, and their arrangement has to be done very carefully. It should depend on the distribution of the lengths of the loading units. As a loading unit must not overlap a section's boundary, there has to be at least one section with the length of the longest loading unit. Let us assume that the number r of different section lengths is given.

The number of sections of each type and their arrangement such that the second objective is minimized is closely related to the product rate variation model for just-in-time scheduling in mass production (see Josefowska 2010) or the apportionment problem as it often arises in indirect democracies. The concepts of the algorithms solving the apportionment problem can easily be adopted in order to determine the number of sections of a certain length and their arrangement. In the following, we will describe these concepts, starting with the determination of the lengths of the section for a given r.

Section lengths

Each section length is denoted by g_j, $j \in \{1, \ldots, r\}$ and we may assume $g_1 < g_2 < \cdots < g_r$. Obviously, there need to be sections of the size of the longest loading units and thus, $g_r = p^{\max}$. The remaining $r - 1$ section lengths can arbitrarily be chosen, but choosing inadequate lengths might lead to a waste of space. "Wasted space" of a loading unit i can be seen as the difference of the length of the smallest section covering the unit's length minus the unit's length:

$$\min_{j \in \{1,\ldots,r\}: g_j \geq p_i} \{g_j\} - p_i$$

As the distribution of the lengths of all loading units is known, the wasted space of each possible loading unit's length can be multiplied with its probability. Thus, we obtain a measure for a set of section lengths $\{g_1, \ldots, g_r\}$. The set of section lengths minimizing the expected waste is chosen for the grid pattern.

Number of sections of a certain length

Assume that the different section lengths $\{g_1, \ldots, g_r\}$ are given. The number of sections of a certain length is determined by a variation of the Hamilton method of apportionment, using the quotas of each section. If p percent of all loading units will

fit to a section of length g_j but not to a smaller one, then (approximately) p percent of all sections should be of length g_j. However, the obtained number of sections might not be integer. Thus, only the integer part is used, and remaining space for further sections is apportioned differently. In such a case, the Hamilton method uses the largest remainder or, equivalently, the value after decimal point as decision criterion. However, this method might not be applicable for the situation at hand as demonstrated in the following example:

Assume that there is only one storage lane of length $L = 100$, and furthermore $r = 2$, $g_1 = 5$, $g_2 = 20$. 3/5 of all loading units will fit to the smaller section. This would imply that $\frac{60}{11} \approx 5.4545$ sections should be of length g_1 and $\frac{40}{11} \approx 3.6363$ sections of length g_2, as only these numbers guarantee that $\frac{3}{5}$th of the sections are of length g_1 and that the whole lane is used ($\frac{60}{11} \times 5 + \frac{40}{11} \times 20 = 100$). Thus, there are (at least) five sections of length g_1, three sections of length g_2 and a remaining space of 15. According to Hamilton method, a third section of length g_2 should be realized, which is not possible without reducing the number of smaller sections.

In order to use remaining space efficiently, we use the following method, which is also based on value after decimal point. A new section of length of the section with the largest value after decimal point will be added, if it still fits into the remaining space, or if only one section of a smaller size has to be removed. Thus, in the example there would be four sections of length g_1 and four sections of length g_2. Removing only one smaller section is not harmful as all loading units fitting into this section also fit into the added bigger one. If this procedure still leaves some space unused, this space is used to evenly enlarge all sections.

Arrangement of the sections

If the number of sections of certain lengths is fixed, they have to be arranged such that the expected total crane rail move is minimized. This can be obtained as follows. Let us firstly assume that we exclusively have to plot the sections of a certain length g_j, $j \in \{1, \ldots, r\}$ on one storage lane. If we limit our attention on the second objective, it would be best to spread these sections evenly along the storage lane. Thus, we get r independent arrangements of sections of a certain length each for a particular section length and a virtual lane, see Fig. 3, representing the example with $g_1 = 5$ and $g_2 = 20$.

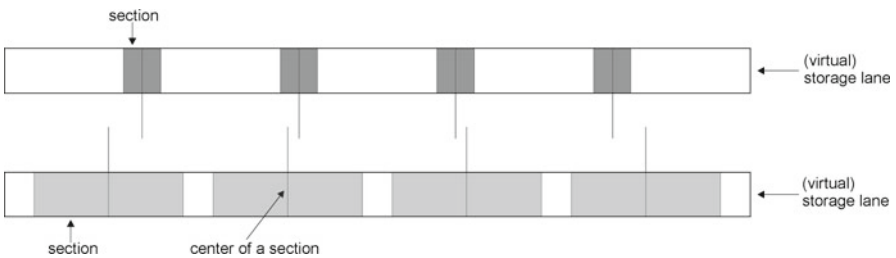

Fig. 3 The preliminary arrangement of sections of two different lengths

Fig. 4 The final arrangement of the sections of two different lengths

The overlay of the virtual lanes and their r arrangements lead to overlapping sections. Overlaps are then resolved by moving the most left section (concerning its center position, denoted by a line in Fig. 3) to the very left end of the storage lane. Afterwards, the next most left section is moved to the left so that it adjoins the first section, and so on. Ties are broken randomly. The resulting arrangement of the example can be seen in Fig. 4.

4 Numerical study

4.1 Instances

From our project partner, we were able to obtain real-life data from a large German transshipment yard. The following data were provided

- The lengths of all loading units that were present in the yard during a 1 day period. There are 18 different lengths of loading units, corresponding to the 23 different loading unit types used in Germany (Kombiverkehr 2008). This data was used as a basis for a distribution of the lengths of all loading units.
- The number of tracks and the train lengths. This bears the expected number of loading units arriving (and departing) in one train bundle. On average, there are 90 units to be unloaded from one train bundle.
- The average dwell time of loading units in the yard's storage area, which is 24 h.
- The average required time for moving one loading unit using a gantry crane, which is 120 s. Note that it is a realistic assumption if each movement is assumed to take exactly 120 s. In a move, most of the time is needed for pick up and drop off, which usually takes fairly constant time. Furthermore, the unit's moves perpendicular to the tracks do not differ much in their length and they can (at least partially) be done simultaneously with crane rail moves.

To simulate the processes of the transshipment yard realistically, our data are completed by:

1. the exact arrival times of trucks, which is modeled as a Poisson process. The expected number of loading units arriving by truck during the unloading process of a train bundle is 45, i.e. half of the number of loading units on a train bundle.
2. the predefined position o_i of a loading unit, which is assumed to be uniformly distributed in the interval $\left[\frac{p_i}{2}, L - \frac{p_i}{2}\right]$. We discretized the values into steps of 1 cm length.
3. the dwell time, which is exponentially distributed with expected value of 24 h.

We obtained uniformly distributed "true" random numbers from the interval [0, 1] using the random.org website. Sample numbers for the various distributions are then generated using inverse transform sampling. We randomly generated 10 instances, each of which corresponds to the unloading of a train bundle.

So far, we have not discussed on how non-empty storage lanes can be simulated. To obtain realistically filled storage lanes, the algorithms repeatedly unload train bundles. In the first repetition, both storage lanes are empty. After the unloading procedure, the amount of time for a loading procedure is estimated by the number of loading units to be moved. Based on the dwell times of all loading units, for each unit it can be determined whether it is still in the storage area and if so, the remaining dwell time can be calculated. Using this "state" of the storage area, time is reset to zero, and the unloading procedure (using the input data of the very same instance) is repeated. We used 10 repetitions of each instance.

4.2 Results

The proposed algorithms have been implemented in Java 2 and run on an Intel® Core™2 Duo, 2.53 GHz PC, with 2.96 GB of memory. The grid pattern algorithm was applied with $r = 1, 2, \ldots, 18$ different section lengths. Each algorithm was to solve 10 repetitions of each of the 10 instances. Both, the rule-based approach and the grid pattern delivered its results within a second.

Table 3 shows the results of all repetitions and all instances. For some values of r, grid pattern performs much better in both, the number of piles and the crane rail distance. The results of the Grid Pattern algorithm are rather straightforward concerning parameter r. $r = 1$ allows an efficient use of sections that become available during processes, but too much space is wasted. With increasing r, less space is wasted at the

Table 3 Cumulative results of 10 repetitions of 10 instances

Algorithm	Number of piles	Crane rail distance (km)
Rule based	2,166	2,178
Grid pattern ($r = 1$)	11,077	2,526
Grid pattern ($r = 2$)	4,746	2,488
Grid pattern ($r = 3$)	2,147	2,148
Grid pattern ($r = 4$)	1,826	2,316
Grid pattern ($r = 5$)	1,685	2,154
Grid pattern ($r = 6$)	1,717	2,314
Grid pattern ($r = 7$)	1,634	2,397
Grid pattern ($r = 8$)	1,634	2,588
Grid pattern ($r = 9$)	1,641	2,437
Grid pattern ($r = 10$)	1,599	2,719
Grid pattern ($r = 11$)	1,696	2,632
Grid pattern ($r = 12$)	1,686	2,625
Grid pattern ($r = 13$)	1,757	2,356
Grid pattern ($r = 14$)	1,794	2,586
Grid pattern ($r = 15$)	1,726	2,567
Grid pattern ($r = 16$)	1,730	2,441
Grid pattern ($r = 17$)	1,835	2,512
Grid pattern ($r = 18$)	1,702	2,529

Table 4 Cumulative results of the last repetition from each of 10 instances

Algorithm	Number of piles	Crane rail distance (km)
Rule based	282	219
Grid pattern ($r = 1$)	1,227	259
Grid pattern ($r = 2$)	553	252
Grid pattern ($r = 3$)	266	221
Grid pattern ($r = 4$)	223	238
Grid pattern ($r = 5$)	207	222
Grid pattern ($r = 6$)	211	237
Grid pattern ($r = 7$)	198	251
Grid pattern ($r = 8$)	191	270
Grid pattern ($r = 9$)	195	250
Grid pattern ($r = 10$)	196	275
Grid pattern ($r = 11$)	207	268
Grid pattern ($r = 12$)	200	264
Grid pattern ($r = 13$)	206	243
Grid pattern ($r = 14$)	214	266
Grid pattern ($r = 15$)	207	262
Grid pattern ($r = 16$)	211	249
Grid pattern ($r = 17$)	223	256
Grid pattern ($r = 18$)	212	260

cost of less flexibility when gaps arise. For the instances at hand, $r = 10$ is optimal concerning piles and $r = 3$ is optimal concerning crane rail moves.

The first repetitions of each instance are less meaningful, because the storage area is not filled on a regular level. Thus, the results of the last repetition are depicted in Table 4. The results in both tables are all about the same. When concerning the stacking objective, $r = 8$ is best, though $r = 10$ delivers a very good solution again. Just like in Table 3, the best solution concerning crane rail moves is obtained for $r = 3$. We may state that if a medium value is chosen for r, the grid pattern can considerably reduce the number of piles compared to the rule-based Approach, which is currently applied at the yard.

4.3 Further tests

Besides the heuristics presented in Sect. 4, we have implemented other ideas, which turned out to be less effective. In the following, we briefly describe these ideas and the according results.

Equal sized grid pattern

At many transshipment yards, the storage lanes are marked with a grid pattern in which all sections have the same length. The only purpose of this grid pattern is to give the

crane operator a better orientation. However, it could also be used for positioning loading units, if we allow loading units to cover more than one section but no section may hold more than one loading unit. We have tested two variants. Firstly, loading units are placed in free sections as close to the predefined position as possible (variant 1). Secondly, in variant 2, loading units are only placed in those free sections, in which one of these sections adjoins an occupied section (similar to the rule-based approach).

When testing variant 1, each storage lane of length 700 m was split into s equal sized sections with s ranging from 100 to 700. The best result was found for 459 sections (of length 1.53 m) with 2,624 piles and 2,056 km crane rail distance. A result, which is only competitive concerning crane rail distance.

Variant 2 corresponds to the rule-based approach if the section length is set to 1 cm. We have tested all sections lengths between 1 and 700 cm. The smallest section length, i.e. the rule-based approach, is best concerning the number of piles, and minimum crane rail distance (2,053 km) is obtained at 556 cm section length. However, this solution causes 8,241 piles.

Weighting functions for states

In this approach, each loading unit is placed such that it is not pilled (if possible). If there are several options, each option is evaluated concerning the resulting state of the storage lanes. The potential of a state is described by the number of further loading units of random length, that can, with a given probability q, be placed without piling, i.e. for a given length distribution of loading units and a given number k of further loading units to be placed, it can be decided whether the probability of placing them without piling is greater than q. If so, we say that the state has $(k, q)\text{-}potential$. We choose the option which leads to $(k, q)\text{-}potential$ with highest k. Only if there are several options, the deviation to the predefined position is minimized.

We have tested this approach using 10 different values for q, $q = 0.1, 0.2, \ldots, 1$. It turned out that increasing q never worsens the results and thus, best results were obtained for $q = 1$. However, with 2,179 piles and 3,253 km crane rail distance, the results are only comparable to the rule-based approach and only concerning the number of piles.

Commercial solvers

SYSP can be linearized (except binary constraints (5) and (6)) and thus, we solved the 10 repetitions of one (static) instance using standard solver Gurobi 4.0 and CPLEX 12. Although we allowed a gap of 0.5%, it took hours to obtain a result, which then turned out to be worst concerning the number of piles and barely comparable concerning crane rail distance.

We furthermore tested a simplified instances of DYSP in which the arrival and departure times of all loading units are known in advance. Even instances with only 15 loading units could not be solved within a time limit of 5 h.

5 Summary and conclusion

We have presented two models SYSP and DYSP for positioning loading units in the storage area of a transshipment yard. Both problems are NP-hard and thus, we evaluated several heuristics. The algorithms were tested on 10 instances, simulating real-life processes at a large German transshipment yard. It turned out that adopting a grid pattern on the storage lanes and placing loading units only in defined sections is most promising.

Future research should focus on the assumptions made in the model. Although neglecting stacking restrictions will most likely not affect the results, considering these restrictions are inevitable for practical application of the proposed methods. Furthermore, an integrated approach considering both, the positioning of loading units in the storage area and crane movements, seems to be promising. A further step would be to analyze the effects of delays while unloading train bundles.

References

Alicke K, Arnold D (1998) Modellierung und Optimierung von mehrstufigen Umschlagsystemen. Fördern und Heben 8:769–772

Borgman B, van Asperen E, Dekker R (2010) Online rules for container stacking. OR Spectr 32:687–716

Bostel N, Dejax P (1998) Models and algorithms for container allocation problems on trains in a rapid transshipment shunting yard. Transp Sci 32:370–379

Boysen N, Fliedner M (2010) Determining crane areas in intermodal transshipment yards: the yard partition problem. Eur J Oper Res 204:336–342

Boysen N, Jaehn F, Pesch E (2010) New bounds and algorithms for the transshipment yard scheduling problem. J Scheduling. doi:10.1007/s10951-010-0200-2

Boysen N, Jaehn F, Pesch E (2011) Scheduling freight trains in rail–rail transshipment yards. Transp Sci 45:199–211

Bruns F, Knust S (2010) Optimized load planning of trains in intermodal transportation. OR Spectr. doi:10.1007/s00291-010-0232-1

Caserta M., Voß S, Sniedovich M (2009) Applying the corridor method to a blocks relocation problem. OR Spectr 31:1–15

Corry P, Kozan E (2006) An assignment model for dynamic load planning of intermodal trains. Comput Oper Res 33:1–17

Corry P, Kozan E (2008) Optimised loading patterns for intermodal trains. OR Spectr 30:721–750

Dekker R, Voogd P, van Asperen E (2006) Advanced methods for container stacking. OR Spectr 28:563–586

Garey M, Johnson D (1979) Computers and intractability: a guide to the theory of NP-completeness. W.H. Freeman and Company, New York

Józefowska J (2012) Just-in-time scheduling in modern mass production environment. In: Ríos-Mercado RZ, Ríos-Solís YA (eds) Just-in-time systems. Springer optimization and its applications, vol 60. Springer, Berlin, pp 171–190

Karp RM (1972) Reducibility among combinatorial problems. In: Miller RE, Thatcher JW (eds) Complexity of computer computations, Plenum Press, New York, pp 85–103

Kim KH, Hong G-P (2006) A heuristic rule for relocating blocks. Comput Oper Res 33:940–954

Kim KH, Park YM, Ryu K-R (2000) Deriving decision rules to locate export containers in container yards. Eur J Oper Res 124:89–101

Kombiverkehr (2008) Frachtbehälter Einteilung nach Längenklassen, UIC

Kozan E (1997) Increasing the operational efficiency of container terminals in Australia. J Oper Res Soc 48:151–161

Montemanni R, Smith D, Rizzoli A, Gambardella L (2009) Sequential ordering problems for crane scheduling in port terminals. Int J Simul Process Model 5:348–361

Souffriau W, Vansteenwegen P, Berghe GV, Oudheusden DV (2009) Variable neighbourhood descent for planning crane operations in a train terminal. In: Sörensen K, Sevaux M, Habenicht W, Geiger MJ (eds) Metaheuristics in the service industry. Lecture notes in economics and mathematical systems, vol 624, Springer, Berlin, pp 83–98

Stahlbock R, Voß S (2008) Operations research at container terminals: a literature update. OR Spectr 30:1–52

Steenken D, Voß S, Stahlbock R (2004) Container terminal operation and operations research: a classification and literature review. OR Spectr 26:3–49