Jena Research Papers in Business and Economics

How to Park Freight Trains on Rail-Rail Transshipment Yards: The Train Location Problem

Michael Kellner, Nils Boysen, Malte Fliedner

02/2010

Jenaer Schriften zur Wirtschaftswissenschaft

Working and Discussion Paper Series
School of Economics and Business Administration
Friedrich-Schiller-University Jena

ISSN 1864-3108

Publisher:

Wirtschaftswissenschaftliche Fakultät
Friedrich-Schiller-Universität Jena
Carl-Zeiß-Str. 3, D-07743 Jena

www.jbe.uni-jena.de

Editor:

Prof. Dr. Hans-Walter Lorenz
h.w.lorenz@wiwi.uni-jena.de

Prof. Dr. Armin Scholl
armin.scholl@wiwi.uni-jena.de

www.jbe.uni-jena.de

How to Park Freight Trains on Rail-Rail Transshipment Yards: The Train Location Problem

Michael Kellner, Nils Boysen, Malte Fliedner

Friedrich-Schiller-Universität Jena, Lehrstuhl für Operations Management, Carl-Zeiß-Straße 3, D-07743 Jena, Germany,
{michael.kellner,nils.boysen,malte.fliedner}@uni-jena.de

Abstract
In modern rail-rail transshipment yards huge gantry cranes spanning all railway tracks allow for an efficient transshipment of containers between different freight trains. This way, multiple trains loaded with cargo for varying destinations can be consolidated to a reduced number of homogeneous trains, which is an essential requirement of hub-and-spoke railway systems. An important problem during the daily operations of such a transshipment yard is the train location problem, which assigns each train of a given pulse to a railway track (vertical position) and decides on each train's parking position on the track (horizontal position), so that the distances of container movements are minimized and the overall workload is equally shared among cranes. For this problem a mathematical model is presented, different heuristic solution procedures are described and tested in a comprehensive computational study. The results show that our procedures allow for a remarkable reduction of train processing time compared to typical real-world train location policies.

Keywords: Railway systems; Transshipment yards; Container handling; Parking positions

1 Introduction

In spite of the extraordinary initiatives of the European Union (EU) and other national authorities, the fraction of the overall freight traffic moved by train decreased from 20% (1970) to 10% (2005) within the last 35 years (EU, 2007). Major handicaps of rail transport are missing the flexibility and a lack of reliability, e.g., caused by the general right of way of passenger traffic in many European countries. Moreover, trains cause high fixed cost, so that freight traffic is only profitable if large trains are moved over long distances. With traditional point-to-point traffic such high demands for freight transport merely exist between large cities, so that areas of lower population density are excluded from freight transport by rail. Only recently, hub-and-spoke systems (which are used in road and air traffic since many years) have been identified as a possibility to profitably move smaller freight volumes (see Ballis and Golias, 2004). In such a system, multiple trains loaded with containers for varying destinations are consolidated to full trains in a hub terminal, which are then moved on long-haul routes to their specific destinations. This way, *economies of transportation* can be realized.

To successfully realize hub-and-spoke railway systems, crane-based rail-rail transshipment yards are required, which allow for an efficient transshipment of containers among trains. Traditional shunting (or classification) yards, where railcars are reshuffled via shunting hills and switches, considerably slow down the transshipment process and, thus, jeopardize on-time deliveries. In modern yards, huge rail-borne gantry cranes spanning the railway tracks move containers between different freight trains, while the railcars remain untouched. Some of these hub yards have already been build in the EU (e.g., Port-Bou at the border between France and Spain, see Martinez et al., 2004) and others are currently under construction (e.g., the German "Mega Hub" in Hannover-Lehrte, see Alicke, 2002; Rotter, 2004) or in design phase. Figure 1 gives a schematic representation of such a rail-rail transshipment yard.

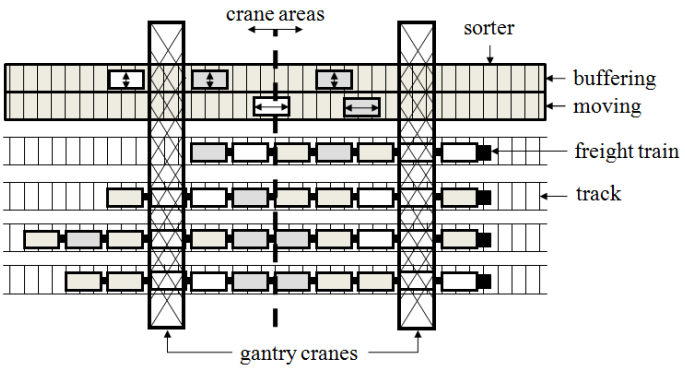

Figure 1: Schematic representation of a rail-rail transshipment yard

A modern rail-rail transshipment yard consists of multiple parallel railway tracks, where successive pulses (one train per track) of freight trains are processed. Container transshipment among trains is conducted by multiple rail-borne gantry cranes spanning

the railway tracks. All cranes share dedicated rail tracks for their horizontal movement along the yard, so that interferences occur whenever an actual container move is blocked by another crane. To eliminate interferences and to avoid cost-intensive online control systems which schedule crane movements, it is a common real-world policy to assign each crane a fixed and separate yard area (see Boysen and Fliedner, 2009). This way, each crane can simply be scheduled by the respective crane operator in a decentralized manner. However, some containers are to be transshipped between railcars which are located in different crane areas. To enable container processing in this case, an additional sorting system is applied, where automated guided vehicles (Bostel and Dejax, 1998) or some rail-mounted vehicles (Alicke, 2002) move and preposition containers. To allow containers overtaking each other within the sorter, a typical system consists of separate moving and buffer lanes (see Rotter, 2004).

In addition to these technical requirements efficient scheduling procedures coordinating the elements of a transshipment yard are required, so that train processing is accelerated. An important decision problem in this context, is to determine the parking positions of trains on the yard. Each train of a given pulse has to be assigned to a track (vertical position) and for each track a parking position (horizontal position) of its respective train must be determined. If trains are parked appropriately, the overall distances of container moves and, thus, processing times of trains can be reduced. Moreover, parking positions determine the division of labor among cranes. As all cranes process trains in parallel, it is the most busy crane which determines the makespan of the current pulse. Thus, the parking positions are to be chosen, so that the maximum workload of cranes is minimized.

The paper on hand investigates the aforementioned train location problem. A mathematical model is presented and different heuristic solution procedures are developed and tested. Furthermore, we compare our optimization approaches to common operational policies typically applied in real-world transshipment yards in a comprehensive simulation of yard operations.

For this purpose the remainder of the paper is organized as follows. In Section 2 existing literature on transshipment yards is reviewed. Then, a detailed problem description with a mathematical model is presented (Section 3) and different heuristic solution procedures are described (Section 4). Section 5 contains the computational part, where the performance of our heuristic solution procedures is tested (Section 5.2) and compared to real-world parking policies in a yard simulation (Section 5.3). Finally, some extensions of the base model are presented (Section 6) and future research needs are summarized (Section 7).

2 Literature Review

Transshipment yards are specific entities in railway systems and especially utilized as part of intermodal container transport. Detailed reviews on railway optimization and intermodal transport are provided, e.g., by Cordeau et al. (1998), Bontekoning et al. (2004), Macharis and Bontekoning (2004) as well as Crainic and Kim (2007). A first

stream of research directly related to transshipment yards deals with strategic problems. For instance, in-depth descriptions of technical and structural properties of transshipment yards are investigated by Ballis and Golias (2002) as well as Rotter (2004). Meyer (1998), Abacoumkin and Ballis (2004), Ballis and Golias (2004) as well as Wiegmans et al. (2006) specifically address the design process of an optimal terminal layout. However, only very few research papers deal with the scheduling problems perpetually arising during the daily operations of a transshipment yards. As the overall scheduling task seems far too complicated to allow for a simultaneous solution, a hierarchical decomposition seems recommendable (see Boysen and Pesch, 2008):

(i) Schedule the succession of trains by assigning them to pulses.

(ii) Decide on the containers' positions on trains.

(iii) Determine fixed crane areas.

(iv) Decide on the trains' vertical and horizontal parking positions.

(v) Decide on the sequence of container moves per crane.

Problem (i) is treated by Boysen and Pesch (2008). Here, a given set of trains to be processed is to be assigned to different pulses. In such a setting, revisits of trains in a later period to receive remaining containers not delivered up to the train's first stay in the yard and double handling of containers are to be avoided. Bostel and Dejax (1998) as well as Corry and Kozan (2006, 2008) treat problem (ii) and provide scheduling procedures to determine the optimal positions of containers on freight trains so that crane moves at the yard are minimized. Fixed crane areas (problem (iii)) are determined by Boysen and Fliedner (2009) as well as Boysen et al. (2009). For a given pulse of trains (with given parking positions) optimal yard areas are determined, so that the maximum workload of cranes is minimized. Problem (v) – the sequencing of crane moves in a transshipment yard – is treated by Alicke (2002). Related problems also occur within seaports (see, e.g., Ng, 2005; Zhu and Lim, 2006; Moccia et al., 2006; Lim et al., 2007; Sammarra et al., 2007).

This paper treats problem (iv) and determines vertical and horizontal parking positions of trains, so that the maximum workload of multiple cranes is minimized. Up to now, only Alicke and Arnold (1998) investigate a train location problem in a transshipment yard. However, they merely model the track assignment of trains (vertical position) in a very basic fashion without, e.g., considering horizontal parking positions, sorter operations and multiple cranes.

3 Detailed Problem Description

The train location problem (TLP) is an operational problem, which has to be solved perpetually for every pulse of trains arriving at a transshipment yard. The TLP assigns each train of the given pulse a track (vertical parking position) and additionally decides

on each train's horizontal parking position along the yard. It is the aim of the TLP to evenly spread the overall workload among the given gantry cranes of the yard, so that by minimizing the maximum workload of cranes train processing of a given pulse is accelerated. This basic decision problem relies on the following premises:

- A rail-rail transshipment yard is typically operated in successive pulses (Bostel and Dejax, 1998) or bundles (Alicke, 2002; Rotter, 2004) of trains, which means that $|G|$ trains (one per track) are simultaneously processed and jointly leave the system not before all container moves are executed, which are required for the respective bundle of trains. Then, another pulse of trains enters the yard. The assignment of trains to pulses is typically conducted in a previous planning step by the respective network operator, because network-wide timetable requirements need to be considered (see Boysen and Pesch, 2008). Thus, it is assumed that the TLP receives the composition of pulses as input data. Note that in Section 6.2 this assumption is relaxed and overlapping service times are assumed. Furthermore, w.l.o.g. we assume that the number of trains equals the number of tracks, which can be easily ensured by inserting "empty" trains.

- Furthermore, it is assumed that all container moves for the given pulse of trains are already specified, so that each move already has a fixed start and target position on its related trains. Train planning, which decides on the location of containers on trains (problem (ii) in Section 2), is a very complex optimization problem by itself, since multiple restrictions, e.g., restrictions on dangerous goods, weight limitations of wagons and train height, need to be considered (see Corry and Kozan, 2006, 2008). Thus, it seems recommendable to solve both optimization problems in a successive manner. We assume that train planning is executed prior to the TLP, so that all container moves are already specified.

- To allow for a better coordination between locomotive engineers and crane operators a typical transshipment yard is subdivided into slots of equal length, whose numbers are painted on the ground along the length of the yard. For instance, German yards typically have a slot length of 14 meters to exactly cover standardized railcars carrying one forty feet (FEU) or two twenty feet (TEU) containers (Boysen and Fliedner, 2009). We presuppose that each freight train can be parked so that any container exactly falls into one slot.

- As gantry cranes share a dedicated rail track for their horizontal movement along the yard, crane interferences threaten. To avoid congestions of cranes, typical real-world yards assign exclusive yard areas to each crane (see Boysen and Fliedner, 2009; Boysen et al., 2009). We assume that these yard areas are already defined, so that TLP receives these crane areas as input data. Note that in Section 6.4 this assumption is relaxed and parking positions and crane areas are simultaneously optimized.

- If start and target position of a container move fall into a single crane's yard area, the move can directly be processed by the respective crane, so that a single pick

and drop operation is required. Such a *direct move* is depicted on the left-hand-side of Figure 2. However, if start and target position are separated by at least one area border a *split move* occurs. The crane which covers the start position has to move the container onto some vehicle waiting in the same slot on the moving lane of the sorter. Then, the container is moved to the slot of its target position, where the second crane (covering the target position) finally moves the container from the sorter to the container's final destination on train. The right-hand-side of Figure 2 depicts the double handling required for such a split move.

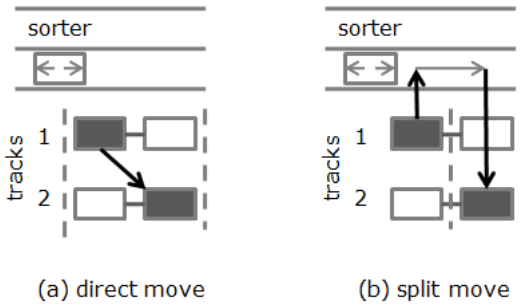

Figure 2: Direct move vs. split move

- Finally, only loaded crane moves which actually carry containers are considered in the optimization, whereas empty crane moves connecting two loaded moves are excluded from approximating each crane's workload. Otherwise, a detailed crane scheduling (problem (v) in Section 2) has to be included into the TLP, which considerably complicates the solution process since two complex optimization problems are to be solved simultaneously. Crane scheduling resembles an asymmetric traveling salesman problem, which is known to be NP-hard in the strong sense (see Garey and Johnson, 1979). With multiple cranes connected via split moves and the sorting system, the problem becomes even more difficult to handle. Thus, to ease the solution process we presuppose that a crane's workload exclusively resulting from loaded moves is strongly positively correlated to its overall workload (consisting of loaded and empty moves), so that minimizing the surrogate objective considerably reduces the underlying objective, as well. This assumption is tested in the computational study of Section 5.3, where the impact of TLP on actually reducing the overall workload is captured in detail.

With these premises on hand, the impact of each train i's parking position p in relation to any other train j's parking position q on the workload of crane c can be stored in a parameter d_{ipjqc} by an appropriate preprocessing, which is clarified by the following example.

Example: Consider a transshipment yard consisting of two tracks and four slots, so that parking positions can be numbered from one to eight: $P = \{1, 2, \ldots, 8\}$. The yard is operated by two cranes, whose distinct yard areas are depicted by the vertical dashed

line in part (a) of Figure 3. The distances among tracks and sorter are also depicted in Figure 3. The given pulse of trains to be processed in the yard consists of two trains, where the container moves between both trains are depicted as arcs in part (a) of Figure 3. The grey box on train 1 represents a railcar, which is to be loaded with a container already waiting in the sorting system. Note that such direct moves to or from the sorter occur, whenever containers are to be transshipped between trains of different pulses.

First, for any train $i \in I$, the set P_i of possible parking positions can be determined. Train 1 can only be parked in the first slot, so that $P_1 = \{1, 5\}$. Otherwise, the train would outreach the yard. Possible parking positions of train 2 amount to $P_2 = \{1, 2, 5, 6\}$. If for both trains, any train's possible parking positions is combined with any other train's positions, eight possible combinations exist. However, since no two trains can take the same positions on a track there are only 4 feasible parking patterns: (1,5),(1,6),(5,1) and (5,2). Two of these patterns are depicted as solutions 1 and 2 in part (b) and (c) of Figure 3, respectively. For a given parking pattern, the resulting workload can be easily determined. Note that cranes can simultaneously be moved in horizontal and vertical direction by applying separate engines, so that it is the maximum distance of both directions which determines each move's length. For instance, the direct move in solution 1 from parking position 1 to 6, covers a horizontal and vertical distance of 14 and 7 meters, respectively, so that the maximum of 14 meters is applied as the moves weight. In total, the resulting workloads for cranes 1 and 2 in solution 1 amount to $d_{11251} = 31$ and $d_{11252} = 26$, respectively. In solution 2, workload parameters result to $d_{11261} = 29$ and $d_{11262} = 24$. A comparison of both solutions reveals that solution 2 with a maximum workload of 29 is preferable. Instead of distances, workload parameters d can also be assigned detailed processing times of cranes which is described in detail within the simulation study of Section 5.3.

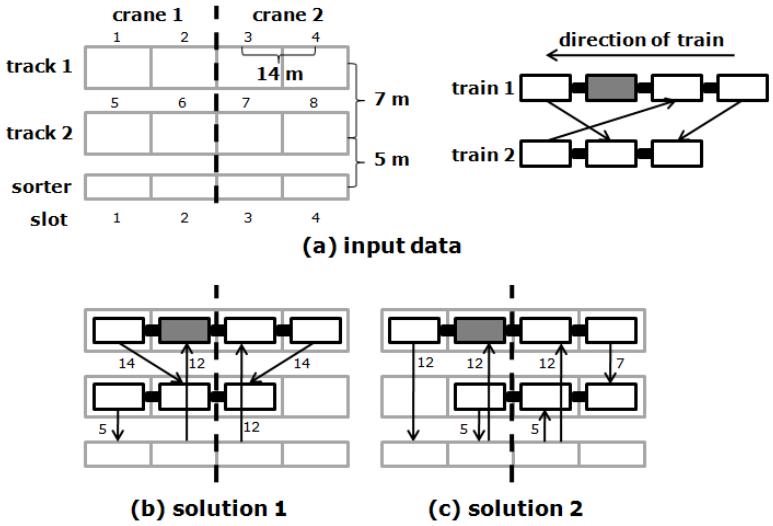

Figure 3: Example: (a) Input data, (b) and (c) solutions

I	set of trains (indices i and j)
C	set of cranes (index c)
G	set of parallel tracks within transshipment yard (index g)
P	set of all parking positions (indices p and q)
P_g	set of parking positions on track g
P_i	set of park positions available for train i
d_{ipjqc}	total distance (or time span) to be processed by crane c between train i at position p and train j at position q
x_{ip}	binary variable: 1, if train i is assigned at position p; 0, otherwise

Table 1: Notation

With suited parameters d_{ipjqc} on hand and the additional notation summarized in Table 1 the train location problem (TLP) can be formalized by a quadratic program consisting of objective function (1) subject to constraints (2) to (4):

$$\text{(TLP) Minimize } F(X) = \max_{c \in C} \left\{ \sum_{i \in I} \sum_{j \in I} \sum_{p \in P_i} \sum_{q \in P_j} d_{ipjqc} \cdot x_{ip} \cdot x_{jq} \right\} \tag{1}$$

subject to

$$\sum_{p \in P_i} x_{ip} = 1 \qquad \forall\, i \in I \tag{2}$$

$$\sum_{i \in I} \sum_{p \in P_g} x_{ip} = 1 \qquad \forall\, g \in G \tag{3}$$

$$x_{ip} \in \{0, 1\} \qquad \forall\, i \in I,\, p \in P_i \tag{4}$$

In objective function (1) the maximum workload of all cranes $c \in C$ is to be minimized, where each crane's workload is determined by summing up the respective workload parameters d_{ipjqc} whenever crane i is parked on position p (with $x_{ip} = 1$) and another crane j is parked on position q (with $x_{jq} = 1$). Equalities (2) ensure that each train i of the given pulse I is assigned to exactly one parking position out of its possible positions P_i. On the other hand, it is to be ensured (constraints (3)) that each track g receives exactly one train, where $P_g \subset P$ comprises all parking positions of track g. Finally, binary variables x_{ip} are defined by (4).

Obviously, TLP with facultative weights d_{ipjqc} is NP-hard in the strong sense, since any instance of the quadratic assignment problem (QAP) can be easily reduced to an instance of TLP with a single crane $C = \{1\}$ where the length of all trains is equal to the length of the yard, so that only one parking position per track is feasible. QAP was proven to be NP-hard in the strong sense by Sahni and Gonzalez (1976).

Algorithm 1: Myopic search procedure (MSP)

```
/* vertical assignment;
order train set I according to descending number of container relations: R =< i_1, i_2, ..., i_|I| ];
assign train i_1 to central track t = ⌈(G + 1)/2⌉;
for j = 1 to ⌈(G − 1)/2⌉ do
    assign train i_2·j to track t − j;
for j = 1 to ⌊(G − 1)/2⌋ do
    relocate train i_2·j+1 to track t + j;
/* horizontal assignment;
assign all trains to first slot s = 1;
while an improvement of the solution was obtained during the last iteration do
    for j ∈ I do
        determine the best possible parking slot s* ∈ {1, ..., S − l_j + 1} in relation to all other currently
        fixed trains;
return track and slot assignment for all trains;
```

4 Algorithms

As the problem is complex, efficient heuristic solution procedures are required for solving problem instances of real-world size. In this work, we present a myopic heuristic procedure and two meta-heuristics, a simulated annealing approach and a genetic algorithm.

4.1 Myopic heuristic procedure

In our myopic search procedure (MSP) assigning vertical and horizontal parking positions is decomposed into two separate steps. In the first step each train is assigned to a track. This assignment decision is guided by the consideration that trains having multiple container relations, i.e., a large number of containers to be processed, are especially promising when located on central tracks. Thus, we order trains according to descending container relations and assign them one after another starting from central track $t = \lceil (G+1)/2 \rceil$ going outwards. In a second step, horizontal parking positions are determined iteratively. Initially, all trains are assigned to the first slot ($s = 1$) at the beginning of the yard. Then, each train is moved individually to its best horizontal position in relation to all other (currently fixed) trains. Clearly, each train can only be moved to feasible slots $s \in \{1, \ldots, S - l_i + 1\}$, where S and l_i denote the overall number of slots and the length (in slots) of train i, respectively. Such a stepwise relocation of the complete train set I is repeated until no improvement of the objective function is obtained. Note that when applying this procedure to the 810 instance of our computational study (see Section 5) on average 3.16 iterations need to be executed before the solution converges and the procedure is stopped. A formal description of MSP is given by Algorithm 1.

Example: Consider the yard layout and train set depicted in Figure 4. Here, three trains are to be parked on a yard having three tracks and four slots. First, MSP sorts trains according to decreasing container relations: $R =< 3, 1, 2]$, where minimum train number is applied as tie-break rule. As train 3 has four container relations, i.e., two containers need to be loaded and unloaded respectively, it is assigned to the central track

2. Then, with all trains parked in slot 1, interim solution 1 on the left-hand side of Figure 5 and a maximum workload of 47 (crane 2) results. After having iteratively evaluated parking positions of trains, the final solution 2 on the right-hand side of Figure 5 with a maximum workload of 45 (crane 2) is returned by MSP.

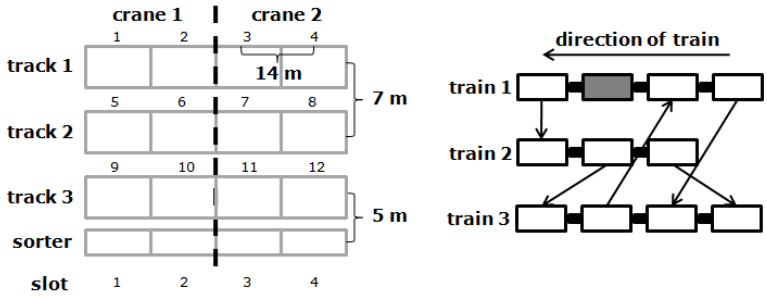

Figure 4: Example with $G = 3$ tracks and $|I| = 3$ trains

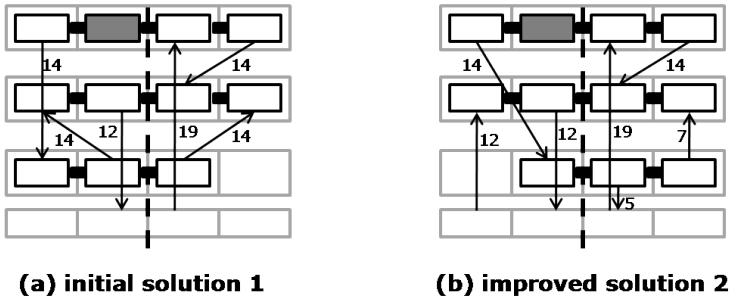

Figure 5: Solutions evaluated by the myopic heuristic procedure

4.2 Simulated annealing

Simulated annealing (SA) is a stochastic local search meta-heuristic that is able to overcome local optima. Specifically, it defines the acceptance of a modified neighboring solution on the basis a probabilistic scheme inspired by thermal processes for obtaining low-energy states in heat baths (e.g., Kirkpatrick et al., 1983; Aarts et al., 1997).

As solution encoding, our SA approach operates on a vector π where elements $\pi_i = (g, s)$ (with $i = 1, \ldots, |I|$) store the current parking position of train i, i.e., consisting of track g and slot s. For our example in Figure 5 the vectors representing solutions 1 and 2 are $\pi(1) = \{(1,1),(3,1),(2,1)\}$ and $\pi(2) = \{(1,1),(3,2),(2,1)\}$, respectively. The initial solution is randomly determined by assigning each train a separate track $g \in G$ and a feasible horizontal slot $s \in \{1, \ldots, S - l_i + 1\}$. Two different neighborhood-functions are applied to modify a current solution, where both kinds of moves are chosen with equal probability:

- A *swap move* interchanges the track assignment of two randomly determined trains, whereas horizontal parking positions remain unchanged. Consider the example of Figure 5. Here, a swap move of trains 2 and 3 would change current solution 2 (with $\pi(2) = \{(1,1),(3,2),(2,1)\}$) to $\pi' = \{(1,1),(2,2),(3,1)\}$.
- Alternatively, a *shift move* does not alter the track assignment of a randomly determined train i, but renews the slot assignment of the train by randomly choosing (with equal distribution) a new feasible horizontal slot out of interval $[1, S - l_i + 1]$. For instance, a shift move could relocate train 2 within solution 2 of Figure 5 to slot 1, so that current solution $\pi(2) = \{(1,1),(3,2),(2,1)\}$ is modified to neighborhood solution $\pi' = \{(1,1),(3,1),(2,1)\}$.

The objective value of any solution vector π can simply be determined by summing the respective weights d_{ijklc} for each crane and identifying the maximum workload over all cranes:

$$F(\pi) = \max_{c \in C} \left\{ \sum_{i \in I} \sum_{j \in I} d_{i,p(g,s)^{\pi_i},j,p(g',s')^{\pi_j},c} \right\}, \tag{5}$$

where $p(g,s)^{\pi_i}$ denotes the parking position of a train i (depending on track g and slot s) in solution vector π. Clearly, a more efficient way to determine these objective values is to merely consolidate diverging weights of those trains i and/or j changed in a neighboring solution. The decision about whether a neighboring solution π' obtained by a move is accepted is decided according to the traditional probability scheme (cf. Aarts et al., 1997):

$$Prob(\pi' \text{ replacing } \pi) = \begin{cases} 1, & \text{if } F(\pi') \leq F(\pi) \\ exp\left(\frac{F(\pi) - F(\pi')}{C}\right), & \text{otherwise} \end{cases}. \tag{6}$$

If accepted, current solution π is replaced by π', so that it becomes the starting point for further local search moves.

Our SA is guided by a simple static cooling schedule (see Kirkpatrick et al., 1983). The initial value for control parameter C is chosen as the objective value of the first random solution ($C^0 = F(\pi^{start})$). In the course of the procedure, C is continuously decreased by multiplying it with a factor of 0.9999 in each iteration. SA is terminated as soon as control parameter C reaches its stop value $C \leq C^0 \cdot 0.001$. Then, the best solution π^* found with minimum objective value $F(\pi^*)$ is returned. Within our computational study, we only report results for the values of control parameters described above. Note that preliminary studies have indicated that this parameter constellation outperforms other settings and has obtained promising results.

4.3 Genetic algorithm

A genetic algorithm (GA) – originally developed by Holland (1975) – is a probabilistic heuristic search procedure, whose solution process emulates evolution in nature. In

analogy to biological evolution, GA operates on a population of individuals, which represent solutions (encoded by a string of genes) for the problem considered. Iteratively, populations are constructed through a number of generations by replacing parent solutions with newly generated child solutions. Following Darwin's concept of "survival of the fittest", each individual's probability of "contributing" to the next generation rises with increasing fitness (better objective value). This is achieved by a probabilistic selection operator, which chooses individuals in proportion to their fitness. Then, using a crossover operator, features of the chosen parent individuals are recombined to form new (child) individuals. Finally, by mutation some properties of an individual are modified randomly. Customizing these basic steps of a GA for TLP is described in the following paragraphs.

Representation and fitness function: The first step in designing a genetic algorithm for a particular problem is to devise a suitable representation scheme. An obvious choice would be to reuse the encoding of our SA approach, where a vector π stores track and slot assignment for each train. However, such a representation suffers from infeasible child solutions during reproduction, which is explained in more detail in the next paragraph. Thus, we apply a slightly modified solution string: Solution vector μ contains elements $\mu_i = (g, f) \, \forall \, i = 1, \ldots, |I|$, which store track assignment $g \in G$ and a floating point number $f \in [0, 1]$ for each train i. Floating number f redefines a horizontal slot assignment as a fraction of each train i's feasible track length, so that the relation between train i's track fraction value $f(\mu_i)$ and its slot assignment $s(\mu_i)$ is defined as follows:

$$s(\mu_i) = \min \left\{ \tau = 1, \ldots, S - l_i + 1 \,\middle|\, f(\mu_i) \leq \frac{\tau}{S - l_i + 1} \right\} \quad \forall \, i \in I. \tag{7}$$

Consider our example of Figure 4. Here, train 2 with a length of three slots is parked in slot 1, if $0 \leq f(\mu_2) \leq 0.5$ holds. If this fraction value is changed to $0.5 < f(\mu_2) \leq 1$, the train moves to slot 2. Once the slot assignment is determined for each train, the fitness value of each individual can easily be calculated by applying equation (5).

Crossover operators: During crossover, bits from each parent string are "combined" to create a child string. The idea is that by creating new strings from substrings of fit parent strings, new and promising areas of the search space will be explored. Many crossover techniques exist in the literature. We chose the widespread one-point crossover with order preservation (also referred to as C1, see Reeves, 1993). Here, a crossover point is randomly drawn out of $[1, |I|]$ and the left-hand side of the first parent is copied to the offspring, with the solution vector being completed by taking in order each element from the second parent. Consider our example problem of Figure 4 and individuals $\mu(a) = \{(1, 0.87), (2, 0.13), (3, 0.41)\}$ and $\mu(b) = \{(3, 0.27), (2, 0.97), (1, 0.34)\}$ being selected for recombination. If position 1 is drawn as crossover point the resulting child solution is $\mu(c) = \{(1, 0.87), (3, 0.27), (2, 0.97)\}$. Note that by combining C1 crossover and our fraction encoding infeasible solutions are avoided with certainty. On the one hand, it is ensured that each track is assigned exactly one train, whereas, for instance, simple

one-point crossover would have led to two trains assigned to track 1 and an empty track 3 for our example. On the other hand, infeasible horizontal positions are eliminated by the fractional encoding. With a direct encoding of each train's slot assignment, C1 crossover would have led to an assignment of infeasible slot 2 to train 3 for our example, whereas the fraction encoding translates $f(\mu(c)_3) = 0.97$ to a feasible assignment of slot 1 for train 3.

Mutation operators: Mutation is generally seen as a guard against loss of valuable genetic information by reintroducing information lost due to premature convergence and thereby expands the search space. Two types of mutation procedure are used in our GA (chosen with equal probability): a swap move, i.e., two randomly drawn trains change track, and a shift move in analogy to the SA procedure. Within the shift move the horizontal parking position of a train is varied by randomly drawing a new fraction value $f \in [0, 1]$. As mutation rate, i.e., the probability of each individual being subject to mutation, preliminary computational tests showed a value of 0.15 as best fitted.

Heuristic improvement operator: By combining the properties of parents to a new solution it is likely that a child solution misses a local optimum. Thus, it is a widespread tradition in the design of meta-heuristics to couple a GA with a local search procedure. We apply such a heuristic improvement operator to the best ten individuals of each generation: First, with given horizontal positions the track assignment of trains is interchanged in a best-fit 2-opt search, which chooses the best track change with improving objective value for a current train, where trains are considered according to decreasing container relations. Then, the horizontal location of each train is iteratively improved as described for MSP (see Section 4.1). The search is aborted if no improvement can be realized within the last iteration.

Parent selection method: Parent selection is the task of assigning an opportunity for reproduction to each individual in the population based on their relative fitness. We decided for a traditional *roulette wheel selection* (see, e.g., Reeves, 1997). Here, to build a new individual (child), two anterior individuals (parents) of the current population will be chosen for reproduction by a probabilistic choice procedure. Each individual has a roulette wheel slot (choice probability) sized in proportion to its fitness value divided by the total fitness over all individuals of the current population. The two chosen individuals and the aforementioned crossover operator are then applied for reproduction.

Population replacement scheme: In order to insert a newly generated child solution into the population, room must be made for the child, which means we must have a method for selecting a population member to be deleted. We implemented different replacement strategies, e.g., applying different ideas of elitism and population overlaps (see Reeves, 1993). However, limited computational experience showed a *tournament replacement* as best fitted. For each child solutions an individual of the current population is randomly drawn (with equal probability). Then, by comparing fitness values of child and anterior individual the better one is identified and (re-)inserted into the population while discarding the other one.

These basic steps are repeated for a population of 200 individuals for 700 generations until the best solution found during the solution process is returned. Within our computational study, we only report results for the values of control parameters described above. Note that promising results could be obtained in preliminary studies where this parameter constellation clearly outperformed other settings.

5 Computational study

In addition to the question for the performance of our solution procedures, the computational study has to answer two other research questions: (i) Is minimizing the maximum workload, which only considers loaded moves (see Section 3), a valid surrogate objective for the actual crane scheduling problem consisting of loaded and empty moves? (ii) What is the advantage of optimized parking positions compared to widespread rules of thumb often applied in real-world yards? However, before answering these questions we elaborate on the setup of our computational study.

5.1 Instance generation and setup of simulation study

To derive test instances for simulating yard operations some assumptions on the yard layout, the container moves to be processed, technical parameters of gantry cranes and the sequencing of crane movements are required. All assumptions are described in detail in the following.

Yard Layout: With regard to the yard layout, we base the study on the typical setting of German transshipment yards. A typical yard length is 700 meters and slots are adjusted to accommodate standard railcars with a total length of 14 meters. Thus, we assume a yard length of $T = 50$ slots, with a horizontal distance of $d^h = 14$ meters between any two adjacent slots. Furthermore, we assume a vertical distance of $d^v = 7$ meters between neighboring tracks and the sorter, which is located below the final track. The number $|G|$ of parallel tracks and the number $|C|$ of gantry cranes are varied as follows: $|G| \in \{4, 6, 8\}$ and $|C| \in \{2, 4, 6\}$, so that differently sized transshipment yards are investigated.

Container moves: The train length (in slots) is randomly determined, where different scenarios (short trains only, long trains only and mixed train lengths) are generated by drawing l_i out of intervals $[15-25]$, $[35-45]$ and $[15-45]$, respectively. Furthermore, we aim to investigate different workload settings, namely low, medium and high workload. Therefore, $\gamma = r \cdot M^{max}$ container moves are determined by randomly drawing trains i and j and train positions k and l (counted from the engine) for each move, where $i \neq j$ must hold. M^{max} and r denote the maximum number of container moves (for given train lengths) and a fraction value randomly drawn from intervals $[0.1, 0.2]$ (low), $[0.4, 0.6]$ (medium) and $[0.8, 0.9]$ (high workload), respectively.

Technical crane parameters: The gantry cranes move in horizontal and vertical direction simultaneously driven by independent engines. In horizontal direction the whole

crane moves on special rail tracks, whereas vertically merely the steeple cab carrying the spreader is moved. Thus, the maximum time span for executing the vertical and horizontal movement determines the processing time of a container move. We assume a velocity of crane and steeple cab of $v^e = 3$ meters per second, if the crane moves empty, whereas the velocity reduces to $v^l = 2$ meters per second, if a container is carried. Once positioned, picking and dropping of containers requires additional processing time. Especially, locating the spreader is precision work, so that we assume a typical time span of $t^d = 45$ seconds for picking or dropping a container. See Alicke (2002) and Martinez et al. (2004) for comparable parameters.

Crane Movement: A typical real-world policy to determine the division of labor among cranes is to partition the yard into equally sized areas (Boysen and Fliedner, 2009). Thus, we follow this widespread approach and assign each given crane the same number of slots (accounting for rounding differences). Note that this assumption is relaxed in Section 6.4. If a bundle of trains is parked and, thus, all container moves are fixed and assigned to gantry cranes, sequencing moves per crane remains a complex optimization problem. In our case, for each crane an asymmetric traveling salesman problem would need to be solved while considering the interdependencies among cranes resulting from split moves. However, the sequence of container moves is typically not optimized by a scheduling procedure but locally determined by the respective crane operator. Thus, to simulate a human decision rule we apply a simple nearest neighbor heuristic. Each crane's starting position is the left hand border of its area, while the steeple cab is positioned over the sorter. From there it consecutively executes the container move closest to its current position. Split moves, e.g., from yard area A to B, are considered by updating crane B's list of unprocessed container moves, not before the respective container arrived in the sorter-segment of crane B. Thus, the list is updated just after crane A processed the first part of the split move (from train into the sorter) and a vehicle (with sorter velocity $v^s = 3$) moved the container into yard area B. With regard to the sorting system it is assumed that vehicles are not a bottleneck and congestions do not occur.

The aforementioned parameters of instance generation are summarized in Table 2. All parameters are combined in a full-factorial design and for each parameter constellation 10 replications are generated, so that $3 \cdot 3 \cdot 3 \cdot 3 \cdot 10 = 810$ different instances were obtained.

symbol	description	values		
$\|G\|$	number of tracks	4	6	8
$\|C\|$	number of cranes	2	4	6
l_i	length of trains	short	long	mixed
		$[15-25]$	$[35-45]$	$[15-45]$
r	fraction of container moves	small workload	medium workload	large workload
		$[0.1,\ 0.2]$	$[0.4,\ 0.6]$	$[0.8,\ 0.9]$

Table 2: Parameters for instance generation

For each of these instances a preprocessing has to be executed, which determines weights d_{ijklc} of total workload resulting from loaded moves for each train pair i and j and feasible parking positions of trains. If feasible positions of a train pair are presumed, the workload w^D of a direct container move from track t and slot s to track t' and slot s' can be determined as follows:

$$w^D((t,s),(t',s')) = 2 \cdot t^p + \max\left\{\frac{|s-s'| \cdot d^h}{v^l}; \frac{|t-t'| \cdot d^v}{v^l}\right\} \tag{8}$$

If processed as a direct move, a container move $((t,s),(t',s'))$ requires a pick and a drop operation $(2 \cdot t^p)$. Additionally, the time for the actual move is to be added, which amounts to the maximum of the crane's horizontal and vertical distance each weighted with velocity v^l (for a loaded move).

Split moves into (with weight w^{IN}) and out of (with weight w^{OUT}) the sorter also require a pick and drop operation and the actual movement time, which is required for the vertical movement between rail track and sorter track t_s:

$$w^{IN}((t,s),(t_s,s)) = 2 \cdot t^p + \frac{(t_s - t) \cdot d^v}{v^l} \tag{9}$$

$$w^{OUT}((t_s,s'),(t',s')) = 2 \cdot t^p + \frac{(t_s - t') \cdot d^v}{v^l} \tag{10}$$

The sum of resulting workload weights w for the respective container moves between two trains amount to weights d_{ijklc}. Note that the workload of empty moves required for the yard simulation can be calculated in the same fashion. Then, the three heuristic solution procedures described in Section 4 are applied to determine parking positions of trains. First, with regard to the solution performance these results are compared to optimal TLP objective values, so that the gap of our heuristic procedures can be determined. Then, resulting parking positions of trains are passed over to our yard simulation, where the schedules of cranes and sorter are determined. This way, the workload of loaded moves (as considered in model TLP) can be compared to the real-world workload (approximated by the simulation), so that the suitability of our surrogate objective and the acceleration of container processing obtained by optimized parking positions can be determined.

5.2 Algorithmic performance

All methods have been implemented in C# 2008 (Visual Studio 2008) and run on an Intel(R) Core(TM) 2Quad 2.83 GHz CPU PC, with 3.23 GB of memory. First, we report the solution performance for our heuristic procedures of Section 4: MSP (myopic search procedure), SA (simulated annealing) and GA (genetic algorithm) compared with optimal solutions gained by a full enumeration of all possible parking patterns of trains. These optimal solutions could be gained for all 270 instances with four tracks ($|G| = |I| = 4$). Table 3 displays these results, where solution quality is represented by the absolute and relative deviations from optimum averaged over all small instances (labeled *avg abs*

and *avg gap* and measured by $F(\delta) - F(O)$ and $\frac{F(\delta)-F(O)}{F(O)} \cdot 100$, where $F(\delta)$ and $F(O)$ are the objective values of our heuristic procedures with $\delta \in \{MSP, SA, GA\}$ and the exact enumeration procedure, respectively). Additionally, maximum deviations (labeled *max abs* and *max gap*) and the number of optimal solutions found (*# opt*) are reported. These quality indicators are also reported for a widespread real-world policy (RWP). In real-world yards, tracks are typically assigned according to a first-come-first-serve policy, which we emulate by a simple random assignment of trains to tracks. Additionally, all trains are typically parked in the first slot.

	RWP	MSP	SA	GA
# opt	0	0	173	178
avg abs	13.70	2.58	0.18	0.18
max abs	48.58	14.02	3.45	3.03
avg gap	68.69	13.07	1.26	0.76
max gap	178.87	55.22	34.53	17.68
avg cpu	< 0.1	< 0.1	1.52	9.63

legend: avg and max abs [minutes],
avg and max gap [%], avg cpu [seconds]

Table 3: Solution performance compared to optimal solutions for all 270 instances with $|G| = 4$

The results of Table 3 reveal a promising solution performance of our heuristic procedures, where, of course, a trade-off exists between runtime and solution quality. Our fastest procedure MSP produces an average gap of 13 % from the optimum, whereas meta-heuristics SA and GA show an average relative deviation of merely 1.26 % and 0.76 %, respectively. On the other hand, solutions time increases from a negligible time span (MSP) to an average of 1.52 (SA) and 9.63 (GA) CPU-seconds (denoted as avg cpu). However, all three heuristic procedures show considerably superior to typical real-world policy RWP. Here, additional maximum workload of cranes amounts to 13.7 minutes on average and the gap averages to a remarkable 68.69 %.

	RWP	MSP	SA	GA
# best	0	41	211	305
avg abs	17.04	3.60	0.75	0.38
max abs	97.14	29.63	7.95	10.27
avg gap	42.16	9.25	2.89	1.35
max gap	173.52	101.65	83.54	61.84
avg cpu	<0.1	< 0.1	2.09	27.86

legend: avg abs and max abs [minutes],
avg gap and max gap [%], avg cpu [seconds]

Table 4: Average solution performance over all 540 test instances with $|G| > 4$

Comparable results are listed in Table 4 for remaining instance with $|G| > 4$, where

optimal solutions could not be gained. Instead, we report the number of best solutions obtained by the respective procedure (# best). Furthermore, absolute and relative deviations are calculated in relation the best solution determined per instance. Deviations are especially low for meta-heuristics SA and GA, whereas solution time increases to 2.09 (SA) and 27.86 (GA) CPU-seconds. However, Figure 6 shows that computational time increases merely linear in the number $|G|$ of tracks for all heuristic procedures. Furthermore, TLP is an operational decision problem solved any couple of hours, so that it seems unproblematic to spend half a minute of computational time.

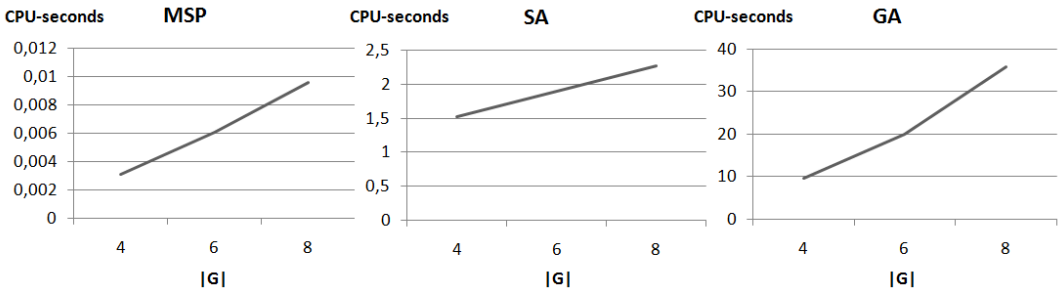

Figure 6: Average solution time (in CPU seconds) of heuristic procedures with varying number $|G|$ of tracks

5.3 Results of yard simulation

In this section we report the results of our yard simulation. Here, parking positions of trains obtained by TLP are passed over and real-world crane and sorter operations are simulated. This way, the results of TLP can be compared with the resulting (approximate) real-world workload of cranes. While TLP minimizes the cranes' workload merely on the basis of loaded moves (surrogate objective, denoted as SURR), the yard simulation provides the actual workload consisting of loaded and empty crane moves (actual objective, denoted as ACT). On average over all 810 instances, the workload of SURR determined by our best-performing GA procedure already makes up 82,81 % of the value of ACT, which results from the overproportional influence of time consuming pick and drop operations. Moreover, the coefficient of correlation (Pearson's product-moment correlation) between both approaches amounts to a remarkable 0.9990. Thus, the conclusion can be drawn, that our surrogate objective of merely considering loaded moves is a suited simplification. On the one hand, the actual objective of reducing the overall workload is strongly supported and, on the other hand, the solution process is considerably alleviated by excluding a detailed crane scheduling.

Furthermore, we aim at investigating the question whether optimized parking positions enable a considerable reduction of train processing time compared to real-world policy RWP. Recall that RWP assigns tracks according to a first-come-first-serve policy and parks all trains in slot 1. As performance measures, we report the average absolute deviation (labeled avg abs) between both policies with regard to the makespan of

processing the current bundle of trains as approximated by our simulation study. Avg abs denominates the acceleration of train processing if improved parking positions are applied instead of RWP in minutes averaged over all instances of the respective parameter constellation. Furthermore, the average relative deviation (labeled avg rel) of both policies in percent is reported, where the deviation is measured by $\frac{F(RWP)-F(\delta)}{F(\delta)} \cdot 100$ with $F(RWP)$ and $F(\delta)$ being the makespan of our yard simulation when parking positions are determined by a real-world rule of thumb or improved with one of our heuristic procedures $\delta \in \{MSP, SA, GA\}$, respectively. Table 5 lists both performance measures in dependency of the parameters: number $|G|$ of tracks and number $|C|$ of cranes, which together reflect the size of a transshipment yard.

		$\|C\|$			
$\|G\|$		2	4	6	total
4	MSP	15.81/32.11	8.78/35.99	6.73/33.23	10.44/33.78
	SA	20.78/51.38	10.33/44.05	8.06/42.30	13.06/45.91
	GA	20.74/51.14	10.23/43.63	8.30/45.22	13.09/46.67
6	MSP	24.31/36.31	15.25/43.59	11.54/45.21	17.03/41.70
	SA	31.84/56.15	17.03/53.19	13.27/54.58	20.71/54.64
	GA	31.87/56.91	17.48/55.33	13.52/57.70	20.96/56.65
8	MSP	33.09/39.28	21.09/47.75	15.73/47.13	23.30/44.72
	SA	42.39/59.94	24.26/59.02	17.85/56.61	28.17/58.52
	GA	43.22/62.49	25.08/64.63	18.84/64.07	29.05/63.73
total	SH	24.40/35.90	15.04/42.54	11.33/41.86	16.93/40.07
	SA	31.67/55.82	17.20/52.09	13.06/51.16	20.64/53.02
	GA	31.94/56.85	17.60/54.53	13.55/55.66	21.30/55.68

legend: avg abs [minutes]/avg rel [%]

Table 5: Absolute and relative speed-up of container processing time depending on yard size

The results reveal a remarkable potential for accelerating train processing. Depending on the size of the yard, possible absolute accelerations (avg abs) of our best-performing GA procedure deviate between 43.22 minutes with eight tracks (high overall workload) and two cranes (low division of labor) and 8.3 minutes with four tracks (low overall workload) and six cranes (high division of labor). Interestingly, the relative acceleration (avg rel) of train processing performs somewhat contrarily. This is explained by the fact, that with a high division of labor the average makespan tends to be lower in value, because a pulse of trains is processed much faster. As a consequence a comparable absolute reduction in makespan leads to a higher relative reduction. In relative terms, train processing is accelerated by between 43.63 % and 64.63 % whenever parking positions are determined by GA.

Further conclusions (in terms of a sensitivity analysis) can be drawn if the speed-up of improved parking positions (determined with GA) is related to the parameters of instance generation. Therefore, Figure 7 displays the average relative deviation (avg rel in %) and

the average absolute deviation (avg abs in minutes) in dependency of the parameters: number $|G|$ of tracks, number $|C|$ of cranes, interval of train lengths l_i and interval of fraction value r, which determines the number of containers (workload) to be processed, respectively.

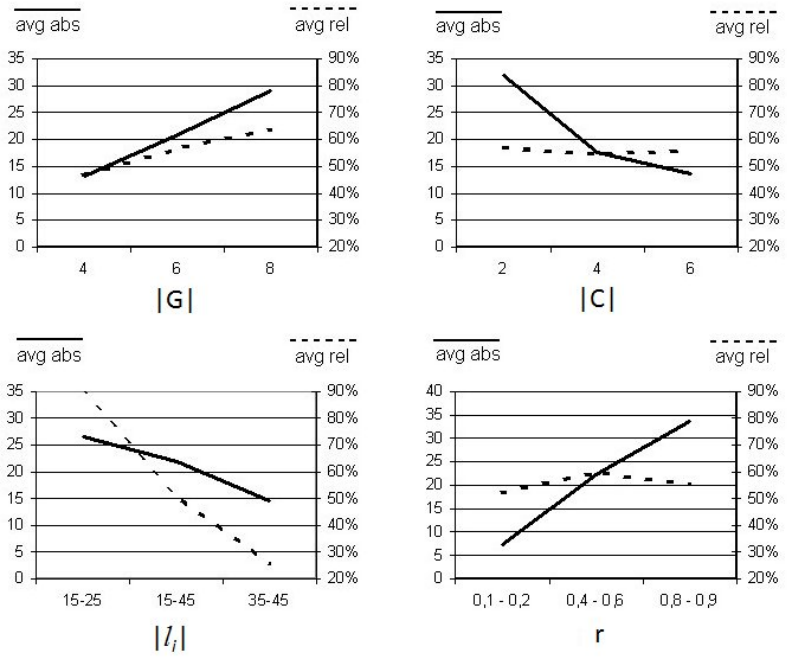

Figure 7: Absolute and relative speed-up in dependency of parameters of instance generation

The following conclusion can be drawn from varying these parameters:

- With an increasing number $|G|$ of tracks the overall workload is increased and, thus, the absolute and relative speed-up of optimal parking positions also rises. The increase of relative speed-up (avg rel) is less distinct, because it is counterbalanced by an overall increasing workload level.
- The higher the division of labor (more cranes $|C|$), the lower absolute (avg abs) speed-ups of optimal parking positions, whereas relative speed-up remains almost unaffected by additional cranes.
- With shorter trains more degrees of freedom exists with regard to finding feasible parking positions within a yard, so that an improvement of parking position becomes even more desirable. Furthermore, with short trains real-world policy RWP, obviously, leads to an unleveled division of labor among cranes. If all trains' horizontal parking positions start in slot one, cranes operating in the front region of

the yard are fully occupied, whereas trains seldom extend into the back of the yard leaving respective cranes idle. Clearly, with longer trains this effect diminishes and improved parking positions lead to less distinct absolute and relative speed-ups.

- With an increasing fraction value r the number of containers to be transshipped rises. Thus, the absolute speed-up of train processing (avg abs) increases, too. However, with regard to the relative speed-up this effect is counterbalanced by an increasing workload level, so that avg rel remains on a constant level.

It can be concluded that a more leveled workload evenly spread over the yard and especially longer trains leaving only a few degrees of freedom for the parking decision reduce the disadvantage of real-world rule of thumb RWP. However, as the results indicate an absolute speed-up of 21.3 minutes (or 55.68%) averaged over all instances optimizing parking positions with one of our procedures can clearly be recommended in a wide range of real-world transshipment yard settings.

6 Extensions of base model

In this section, extensions of the basic parking problem are considered by defining resulting model modifications and lining out the necessary modifications for solving these problems with the solution procedures of Section 4. Especially, we investigate (i) the rotation of trains, (ii) overlapping service times of trains, (iii) multiple trains per track and (iv) the integration of optimizing crane areas.

6.1 Rotation of trains

If respective holding and switching tracks are available in a transshipment yard (see, e.g., Ballis and Golias, 2002, for the track layout of terminals), trains may enter transshipment tracks bi-directionally. In this case, the direction of each train becomes an additional decision variable. This decision can be integrated in our basic TLP model of Section 3 by substituting constraints (2), which assign each train exactly one parking position, with:

$$\sum_{p \in P_i} x_{ip} + \sum_{q \in P_j} x_{jq} = 1 \quad \forall\, (i,j) \in \Omega. \tag{11}$$

Here, it is assumed that each real-world train is duplicated (along with its respective weights d), so that a separate train is stored for each direction within train set I. Furthermore, set Ω contains all pairs of trains (i,j) representing the same real-world train. With these adoptions of the input data, constraints (11) ensure that each real-world train can be parked in either direction.

Clearly, the additional decision on each train's direction is to be integrated into the solution vectors of our SA and GA procedures. Specifically, the binary information k_i ($k_i = 0$: train i parks with engine in front and $k_i = 1$ engine at last) stored with each train in the solution vector and an additional flip move, which changes the direction of a

randomly chosen train, are required to extend our procedures. The latter can be applied as an additional neighborhood function within SA and as a mutation operator within GA, respectively.

6.2 Overlapping service times of trains

Typically, terminal operators aim at processing trains in disjunct bundles of trains (see Section 3), so that a rapid and direct exchange of containers is enabled. However, terminal policies might also allow for varying and, thus, overlapping service times of trains (see Rotter, 2004). As the assignment of service times to trains is typically decided by the network operator in a prior decision step (see Section 3), we presuppose given arrival times a_i and departure times d_i for each train i. Note that while planning these service times it is to be ensured, that the joint service window of interrelated trains is long enough to transship all containers to be exchanged between the respective trains. Within TLP it is to be ensured that trains simultaneously processed in the yard may not be assigned the same track, which can easily be ensured by exchanging constraints (3) with:

$$\sum_{p \in P_g \cap P_i} x_{ip} \cdot \sum_{q \in P_g \cap P_j} x_{jq} \cdot (d_j - a_i) \cdot (d_i - a_j) \leq 0 \quad \forall\, g \in G;\, i, j \in I \text{ with } i \neq j. \tag{12}$$

For solving TLP with overlapping service times an adoption for our heuristic MSP procedure is readily available. In addition to the successive steps of (i) assigning a track and (ii) a slot to each train as described for basic TLP, in an initial step a feasible train schedule needs to be determined. Generating a feasible schedule is formally defined by Algorithm 2. Here, after sorting all trains by departure time, trains are successively assigned to tracks. Any train is assigned to an available track with latest departure time of currently assigned trains. If no track is available, no feasible solution exists for the problem instance (see Ding et al., 2005, for a similar approach for a related problem). Such a feasible train schedule is then passed over to the aforementioned steps of MSP. First, the tracks are interchanged (together with all trains assigned) by sorting them according to decreasing container relations and reassigning them to tracks, i.e., starting from the central track going outwards. Then, in a final step each train is successively moved into its best horizontal parking position in relation to all other currently fixed trains until no improvement of the objective function can be obtained. Note that TLP with overlapping service times is closely related to the gate assignment problem (e.g., see Dorndorf et al., 2007), where airplanes with given service times need to be assigned to gates of an airport, so that walking distances of passengers are minimized. Thus, for adopting our SA and GA approaches the neighborhood functions and moves, e.g., defined by Ding et al. (2005) can directly be applied.

6.3 Multiple trains per track

If the operational policy of a yard allows multiple trains to be parked per track, it is to be ensured that trains sharing a track do not overlap with regard to their horizontal

Algorithm 2: Finding a feasible schedule for TLP with overlapping service times

order train set I according to earliest departure times d_i: $R =< i_1, i_2, \ldots, i_{|I|}]$;
initialize t_g (current availability time of track g): $t_g = -1 \, \forall \, g \in G$;
for *each train* $i \in R$ **do**
    find track $g \in G$ such that t_g is maximized among those tracks for which $t_g < a_i$ holds;
    if such track g exists, assign train i to track g and update $t_g = d_i$;
    if no track g can be found, a feasible solution for the problem instance does not exist;
return track assignment for all trains;

parking positions. To integrate this extension within TLP, constraints (3) which ensure that exactly one train is parked per track are to be substituted with:

$$x_{ip} \cdot (s(p) + l_i) < x_{jq} \cdot s(q) + M \cdot (1 - x_{jq}) \quad \forall \, g \in G; \, i, j \in I; \, p, q \in P_g \text{ with } s(p) < s(q), \tag{13}$$

where $s(p)$, l_i and M denote the slot of parking position p, the length (in slots) of train i and a big integer, respectively. Note that, for instance, $M = |P_g|$ can be chosen. These inequalities ensure that for each pair of trains i and j, if parked on the same track g with i on a prior slot than j, the ending slot of train i (start-up position plus train length) is located prior to train j's initial position.

Proposition: With the extension of allowing multiple trains per track even finding a feasible solution for TLP becomes NP-hard in the strong sense.

Proof: As a straightforward reduction from the 3-Partition problem is available, which is well known to be NP-hard in the strong sense (see Garey and Johnson, 1979), we only define the reduction scheme and sketch the proof. Given an instance P of 3-Partition defined by $3q$ positive integers a_i $(i = 1, \ldots, 3q)$ and a positive integer B with $B/4 < a_i < B/2$ and $\sum_{i=1}^{3q} a_i = qB$, does there exist a partition of the set $\{1, 2, \ldots, 3q\}$ into q sets $\{A_1, A_2, \ldots, A_q\}$ such that $\sum_{i \in A_j} a_i = B \, \forall j = 1, \ldots, q$? The instance P' of TLP with multiple trains per track is defined as follows: $|I| = 3q$, $|G| = q$, $|P_g| = B \, \forall \, g \in G$ and $l_i = a_i \, \forall \, i \in I$. Clearly, any solution of P can be transformed into a solution for P' by simply assigning each set of three elements to a separate track and vice versa. In any solution for P' exactly three trains are parked per track due to the restriction of integer values $B/4 < l_i = a_i < B/2$. Moreover, any of the B slots per track must be fully occupied by these three trains because an empty slot on one track inevitably causes another overloaded track, which is impossible in a feasible solution due to each track's restriction in length. $\square$

As determining feasible solutions is no trivial task anymore, infeasible solutions, where trains overlap or outreach the yard, are to be accepted within our solution procedures. When calculating the fitness of these infeasible solutions penalty costs need to be integrated into the objective function, which penalize solutions proportionally the their degree of infeasibility. Respective approaches, which are well-known extensions for sim-

ulated annealing (e.g., Reeves, 1993) and genetic algorithms (e.g., Goldberg, 1989), can be easily integrated into our solution approaches.

6.4 Integration of optimized crane areas

Up to now, it was presupposed that the yard is equally shared among cranes, so that each crane receives its dedicated and equally sized yard area. Recall that the policy of dedicated yard areas is a widespread real-world policy to avoid crane interferences (see Boysen and Fliedner, 2009). However, dedicated crane areas must not be equally shared but can variably be sized according to the workload of the current bundle of trains. For any given assignment of trains to parking positions optimal yard areas can efficiently be determined in polynomial time, for instance by a Dynamic Programming (DP) approach (see Boysen et al., 2009). This DP approach divides the solution process into $c = 1, \ldots, |C|$ stages, where each stage represents a crane c (numbered from left to right). The basic formula:

$$f_{cs} = \min_{c-1 \leq j \leq s-1} \{\max \{f_{c-1j};\, W(j+1, s)\}\} \qquad \begin{array}{l} \forall c = 2, \ldots, |C|-1,\ s = c, \ldots, c+S-|C| \\ \text{and } \ c = |C|,\ s = S \end{array}, \tag{14}$$

recursively determines optimal partial objective values f_{cs} for any crane c and feasible right-hand border slot s, where current workload $W(j+1, s)$ comprises all moves (direct plus split moves) falling in the current area reaching from slot $j+1$ to slot s. Figure 8 depicts the DP-graph and the resulting yard partition for our example of Figure 3 given the parking parking positions of solution 2 (depicted as part (c) in Figure 3) and three cranes. The maximum workload amounts to 24 and is processed by crane 3.

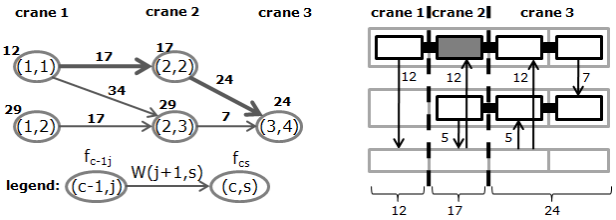

Figure 8: DP-graph and yard partition for three cranes

Clearly, TLP and the yard partition problem (YPP) are heavily interdependent. TLP determines parking positions under given crane areas, whereas YPP partitions the yard into crane areas for a bundle of trains with given parking positions. Thus, further accelerations of train processing might be gained if both problems are solved in an integrative fashion. To evaluate this potential the following computational experiment is carried out. For a representative parameter constellation ($|G| = 4$, $|C| = 4$, $l_i \in [35, 45]$ and $r \in [0.4, 0.6]$) 100 instances were generated as described within Section 5.1. Then, these instances are solved by alternately solving TLP (with GA) and YPP (with DP), where the results of TLP (parking positions) are passed over to YPP (yard partitions) and vice

versa, until no improvement of train processing time (determined by the yard simulation) could be obtained. Our integrative procedure is initialized with equally sized crane areas. Such an integration of TLP and YPP leads to an average (absolute) speed-up of another 7.42% (9.4 minutes) compared to an isolated execution of TLP (with equal crane areas). Note that the integrative approach converges quickly as on average over all 100 instances merely 3.1 iterations are executed, which means that on average one instance of TLP and YPP are solved before no improvement can be realized.

7 Conclusions

This paper investigates the train location problem, which decides on vertical and horizontal parking positions of trains in a rail-rail transshipment yard, so that the workload is evenly shared among gantry cranes and the makespan of train processing is minimized. The problem is formalized by a mathematical program and different heuristic solutions procedures are presented. A comprehensive computational study reveals a promising solution performance of these procedures. Furthermore, a simulation of real-word yard operations shows a tremendous potential for accelerating container processing, when comparing our procedures with typical real-world rules of thumb. Depending on the specific yard setting and workload between 8 and 43 minutes (44–65 %) of processing time per bundle of trains can be saved.

There are several ways in order to build up on this study in future research. On the one hand, further solution procedures could be developed. Especially, exact solution procedures would be a valuable contribution as real-world problem dimensions (up to 8 tracks, 6 cranes and 50 slots) seem within reach for efficient exact procedures. On the other hand, a related parking problem of trains also exists in conventional rail-road transshipment yards, which are applied in intermodal transport. In these yards, gantry cranes transship containers between freight trains and truck (and vice versa) and, clearly, parking positions of trains heavily affect the workload of cranes in such a setting as well.

References

[1] Aarts, E.H.L., Korst, J.H.M., van Laarhoven, J.M., 1997. Simulated Annealing, In: Aarts, E.H.L., Lenstra, J.K. (eds.) Local search in combinatorial optimization, Chichester et al., 91–120.

[2] Abacoumkin, C., Ballis, A., 2004. Development of an expert system for the evaluation of conventional and innovative technologies in the intermodal transport area, European Journal of Operational Research 152, 410–419.

[3] Alicke, K., 2002. Modeling and optimization of the intermodal terminal Mega Hub, OR Spectrum 24, 1–17.

[4] Alicke, K., Arnold, D., 1998. Optimierung von mehrstufigen Umschlagsystemen, Fördern und Heben 8, 769–772.

[5] Ballis, A., Golias, J., 2002. Comparative evaluation of existing and innovative railroad freight transport terminals, Transportation Research Part A 26, 593–611.

[6] Ballis, A., Golias, J., 2004. Towards the improvement of a combined transport chain performance, European Journal of Operational Research 152, 420–436.

[7] Bontekoning, Y.M., Macharis, C., Trip, J.J., 2004. Is a new applied transportation research field emerging? A review of intermodal rail-truck freight transport literature, Transportation Research Part A 38, 1–24.

[8] Bostel, N., Dejax, P., 1998. Models and algorithms for container allocation problems on trains in a rapid transshipment shunting yard, Transportation Science 32, 370–379.

[9] Boysen, N., Fliedner, M., 2009. Determinig crane areas in intermodal transshipment yards: The Yard Partition Problem, European Journal of Operational Research 204, 336–342.

[10] Boysen, N., Fliedner, M., Kellner, M., 2009. Determining fixed crane areas in rail-rail transshipment yards. Jena Research Papers in Business and Economics (JBE) 07/2009, FSU Jena Germany.

[11] Boysen, N., Pesch, E., 2008. Scheduling freight trains in rail-rail transhipment yards, Jena Research Papers in Business and Economics (JBE) 11/2008, FSU Jena Germany.

[12] Cordeau, J.-F., Toth, P., Vigo, D., 1998. A survey of optimization models for train routing and scheduling, Transportation Science 32, 380–404.

[13] Corry, P., Kozan, E., 2006. An assignment model for dynamic load planning of intermodal trains, Computers & Operations Research 33, 1–17.

[14] Corry, P., Kozan, E., 2008. Optimised loading patterns for intermodal trains, OR Spectrum 30, 721–750.

[15] Crainic, T.G., Kim, K.H., 2007. Intermodal transport, In: Barnhart, C., Laporte, G. (eds.) Transportation, Handbooks in Operations Research and Management Science 14, 467-538.

[16] Dorndorf, U., Drexl, A., Nikulin, Y., Pesch, E., 2007. Flight gate scheduling: State-of-the-art and recent developements, Omega 35, 326–334.

[17] Ding, H., Lim, A., Rodrigues, B., Zhu, Y., 2005. The over-constrained airport gate assignment problem, Computers & Operations Research 32, 1867–1880.

[18] Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness, W.H. Freeman, San Francisco.

[19] Holland, J.H., 1975. Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor.

[20] Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing, Science 220, 671–680.

[21] Lim, A., Rodrigues, B., Xu, Z., 2007. A m-parallel crane scheduling problem with a non-crossing constraint, Naval Research Logistics 54, 115–235.

[22] Macharis, C., Bontekoning, Y.M., 2004. Opportunities for OR in intermodal freight transport research: A review, European Journal of Operational Research 153, 400–416.

[23] Martinez, F.M., Gutierrez, I.G., Oliveira, A.O., Bedia, L.M.A., 2004. Gantry crane operations to transfer containers between trains: A simulation study of a Spanish terminal, Transportation Planning and Technology 27, 261–284.

[24] Meyer, P., 1998. Entwicklung eines Simulationsprogramms für Umschlagterminals des Kombinierten Verkehrs, Ph.D. thesis, Universität Hannover.

[25] Moccia, L., Cordeau, J.F., Gaudioso, M., Laporte, G., 2006. A branch-and-cut algorithm for the quay crane scheduling problem in a container terminal, Naval Research Logistics 53, 45–59.

[26] Ng, W.C., 2005. Crane scheduling in container yards with inter-crane interference, European Journal of Operational Research 164, 64–78.

[27] Reeves, C.R., 1993. Modern heuristic techniques for combinatorial problems, Blackwell Scientific Publications Oxford.

[28] Reeves, C.R., 1997. Genetic algorithms for the operations researcher, INFORMS Journal on Computing 9, 231–250.

[29] Rotter, H., 2004. New operating concepts for intermodal transport: The Mega Hub in Hanover/Lehrte in Germany, Transportation Planning and Technology 27, 347–365.

[30] Sahni, S., Gonzalez, T., 1976. P-complete approximation problems, Journal of the Association of Computing Machinery 23, 555–565.

[31] Sammarra, M., Cordeau, J.F., Laporte, G., Monaco, M.F., 2007. A tabu search heuristic for the quay crane scheduling problem, Journal of Scheduling 10, 327–336.

[32] Wiegmans, B.W., Stekelenburg, D.T., Versteegt, C., Bontekoning, Y.M., 2006. Modeling rail-rail exchange operations: An analysis of conventional and new-generation terminals, Transportation Journal 2006(3), 5–20.

[33] Zhu, Y., Lim, A., 2006. Crane scheduling with non-crossing contraints, Journal of Operational Research Society 57, 1472–1481.