Proceedings of the Federated Conference on Computer Science and Information Systems pp. 227–231
DOI: 10.15439/2018F26
ISSN 2300-5963 ACSIS, Vol. 15

A Graph-Theoretic Approach to the Train Marshalling Problem

Jens Dörpinghaus
Fraunhofer Institute for Algorithms and Scientific Computing,
Schloss Birlinghoven, Sankt Augustin, Germany
Email: jens.doerpinghaus@scai.fraunhofer.de

Rainer Schrader
Institut für Informatik,
Universität zu Köln, Germany
Email: schrader@zpr.uni-koeln.de

***Abstract*—Rearranging cars of an incoming train in a hump yard is a widely discussed topic. We focus on the train marshalling problem where the incoming cars of a train are distributed to a certain number of sorting tracks. When pulled out again to build the outgoing train, cars sharing the same destination should appear consecutively. The goal is to minimize the number of sorting tracks. We suggest a graph-theoretic approach for this $\mathcal{NP}$-complete problem. The idea is to partition an associated directed graph into what we call pseudochains of minimum length. We describe a greedy-type heuristic to solve the partitioning problem which, on random instances, performs better than the known heuristics for the train marshalling problem.**

I. Introduction

A HUMP yard usually consists of a hump and a set of classification or sorting tracks and one or more roll-in and pull-out tracks [1]. In hump yards freight cars are arranged or rearranged into a specific sequence of cars. The outgoing trains will deliver goods to new destinations. A practical introduction to hump yards with examples can be found in the work of Hiller [2].

This can be very complex. For example the hump yard in in Zürich-Limmattal (CH) consists of 18 roll-in tracks, 64 sorting tracks with a length of 650-850 meters and 16 roll-out tracks, see [2][3].

Every incoming car arriving at the hump yard will be assigned to a sorting track. At the end of this process all cars of every sorting track will be placed as a block on the roll-out track. For an optimization approach the number and length of sorting tracks, the number of roll-in and pull-out operations can be minimized.

Hansmann provided a general class of *Sorting of rolling Stock Problems* (SRSP) in [4]. We will focus on the *Train Marshalling Problem* (TMP): using a minimum number of tracks, rearrange the cars in a hump yard in such a way that cars sharing the same destinations appear consecutively in the rearranged train.

During the process only two movements are allowed: the sorting of cars to the tracks and one pull-out movement for all cars. The tracks are not limited in length, so we can think of the tracks as stacks. We only allow one roll-in operation per car and one pull-out operation per track. No further shunting is allowed.

Apparently, TMP was first introduced by Zhu and Zhu [5] in 1983 who considered it under additional constraints and gave first results and polynomial algorithms. In 2000, Dahlhaus et al. [6] proved that TMP is $\mathcal{NP}$-complete and introduced new bounds. Brueggeman et al. show in [7] that the problem is fixed parameter tractable. In another work by Dahlhaus, Manne, Miller and Ryan [8] they described similar problems. More bounds and algorithms can be found in the work of Beygang [9] and Beygang et al. [1]. They introduced a graph-theoretic approach by considering the interval graph of a given instance. The problem also occurs in the works of Hansmann [4]. Other approaches can be found in the work of Rinaldi and Rizzi [10] who focused on dynamic programming and Haahr and Lusby [11].

First of all we will give a short formal problem description and all relevant definitions. After introducing pseudochains and discussion splittable destinations we will derive a novel greedy heuristic to solve the TMP. We will evaluate the results on some random instances and finish with a conclusion.

II. Problem description

With every *car* i in the hump yard we associate a natural number $\sigma_i \in \mathbb{N}^+$ representing the destination of the car. A *train* σ of length n then is a sequence

$$\sigma = (\sigma_1, ..., \sigma_n)$$

of cars with $\sigma_i \in \{1, ..., d\}$ for $i \in \{1, ...n\}$.

Example II.1. *Let $\sigma = (1, 2, 1, 3, 2)$. There are three destinations, where the first and third car and, resp., the second and the last have the same destination.*

We want to rearrange the cars in a departing train such that all cars are sorted in blocks according to their destination. For this, only two shunting operations are permitted: the roll-in movement of a car to one of the sorting tracks and the pull-out of all cars on a sorting track. The goal is to minimize the number of sorting tracks, denoted by $K(\sigma)$. Since only one shunting operation per sorting track is allowed the minimization of shunting operations is equivalent to the minimization of sorting tracks.

For a given sequence σ let S_k be the elements of σ with destination k. Then we may describe the incoming sequence by a partition $S = \{S_1, \ldots, S_d\}$ of $\{1, 2, ...n\}$. Dahlhaus et al.

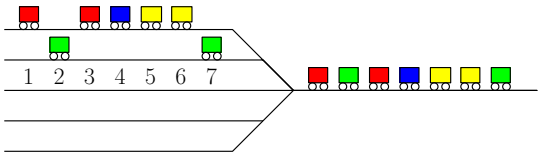

Fig. 1: Illustration for example II.2.

[6] have shown that TMP may be rephrased as follows: find the smallest number $K(S)$ and a permutation π of $1,\dots,d$ such that the sequence of numbers

$$\underbrace{1,2,...,n,1,2,...,n,1,2,...,n}_{K(S)\text{-times}}$$

contains the elements of $S_{\pi(1)}$ as a subsequence followed by all elements of $S_{\pi(2)}$ and so on.

Example II.2. *Let $n = 7$, $d = 4$ und $S = \{S_1, S_2, S_3, S_4\}$ with $S_1 = \{1,3\}$, $S_2 = \{2,7\}$, $S_3 = \{4\}$ and $S_4 = \{5,6\}$. Then $K(S) = 2$ and $\pi(1,3,4,2)$, see figure 1 for an illustration:*

$$\underbrace{\mathbf{1}\,2\,\mathbf{3}}_{S_1}\ \underbrace{\mathbf{4}}_{S_3}\ \underbrace{\mathbf{5}\,\mathbf{6}}_{S_4}\ 7\ 1\ \underbrace{\mathbf{2}\,3\,4\,5\,6\,\mathbf{7}}_{S_2}$$

We now define some necessary preliminaries following the work of Beygang in [9].

III. Preliminaries

Let $\mathbb{S}^n$ be the set of all problem instances of TMP with n cars. For $S \in \mathbb{S}^n$ let $d = d(S)$ be the number of destinations in this instance. For an instance $S \in \mathbb{S}^n$, a *track assignment* is function $tr : \{1,...,n\} \to \mathbb{N}$ which assigns a track to every car. A track assignment is feasible if it gives a feasible solution for TMP.

For a given sequence σ and a destination k let $first(k)$ denote the position of the first occurrence of k and $last(k)$ be its last occurrence. Let $I_k = [first(k), last(k)]$ be the associated interval. Then the intervals induce a partial order on the set of destinations via $i < j$ if $last(i) < first(j)$. We consider the associated comparability graph and its complement, the interval graph.

Definition III.1. *(Comparability Graph associated with an Input Instance) For a given instance S of TMP, the associated comparability graph graph is given by $D(S) = (V, A)$ such that $V = (I_1, ..., I_d)$ and $(I_k, I_j) \in A$ if $k < j$.*

Definition III.2. *(Interval Graph associated with an Input Instance) For a given instance S of TMP, the associated interval graph is given by $G(S) = (V, E)$ such that $V = (I_1, ..., I_d)$ and $(I_k, I_j) \in E$ if $I_k \cap I_j \neq \emptyset$.*

Beygang already introduced some bounds and two important heuristics for the TMP. The deterministic SPLIT-Algorithm was introduced in [9] and computes a feasible solution by splitting destinations whenever possible in $O(n)$. The GREEDY-Algorithm was also introduced in [9]. It finds a feasible solution by partitioning the interval graph $G(S)$ into a minimum number $\chi(G_S)$ of stable sets, each assigned to one track. Recall that this is equivalent to partitioning $D(S)$ into a minimum number of chains. We will generalize this approach to partition $D(S)$ into pseudochains.

IV. Pseudochains

Let $D = (V, A \cup B)$ be a directed graph with a set B of blue arcs, $A \cap B = \emptyset$. We allow that $B = \emptyset$ or $A = \emptyset$. Recall that a chain in a transitively oriented graph is a subset $v_1, \dots, v_k$ of vertices such that $(v_i, v_j) \in A$ for all $1 \leq i < j \leq k$.

Definition IV.1. *(Pseudochain) Let $D = (V, A \cup B)$ as above such that the subgraph D_A induced by the arcs in A is transitively orientable. $\mathcal{C} \subseteq V$ is a* pseudochain *of length $\ell(\mathcal{C}) = k \geq 1$ if $\mathcal{C}$ can be written as*

$$\mathcal{C} = C_1, b_2, C_2, b_3, C_3, ..., b_k, C_k,$$

where the C_i's are mutually disjoint chains in D_A with last element a_i and first Element c_i and $(a_{i-1}, b_i), (b_i, c_i) \in B$ for $2 \leq i \leq k$.

Figure 2 illustrates a pseudochain of length three.

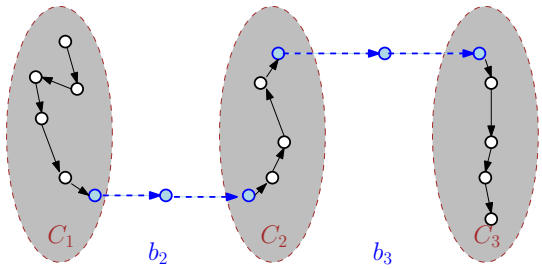

Fig. 2: A pseudostable chain of length 3. Transitive arcs are omitted, dashed arcs correspond to blue arcs.

Now we can define the minimization problem as follows.

Definition IV.2. *(minPC) Given a directed graph $D = (V, A \cup B)$ with a set B of blue edges, $A \cap B = \emptyset$ such that D_A is transitively orientable. Partition V into pseudochains $P = \mathcal{C}_1, \dots, \mathcal{C}_k$ such that the total length $\ell(P) = \sum_{i=1}^{k} \ell(\mathcal{C}_i)$ of the partition is minimal.*

Lemma IV.3. *Given a directed graph $D = (V, A \cup B)$ with a set B of blue edges such that D_A is transitively orientable and a minimum partition V into pseudochains $P = \mathcal{C}_1, \dots, \mathcal{C}_k$. Then there is a partition P' with $\ell(P') = \ell(P)$ such that for all centers $c(b_i)$ of all blue paths b_i in P' exist two nodes u in C_{i-1} and v in C_i so that $(c(b_i), v)$, $(u, c(b_i))$ and $(u, v) \notin A$.*

V. Splittable destinations

Observe that we can always produce a feasible track assignment by opening a track for each destination. But we may be able to do better by distributing the cars of one destination to two tracks. For this, we consider three destinations (a, b, c) with $I_a \cap I_b \neq \emptyset$ and $I_b \cap I_c \neq \emptyset$. Thus we need at least two tracks for $S(a) \cup S(b)$ and two tracks for $S(b) \cup S(c)$. We call the destination b *splittable* with predecessor a and successor c

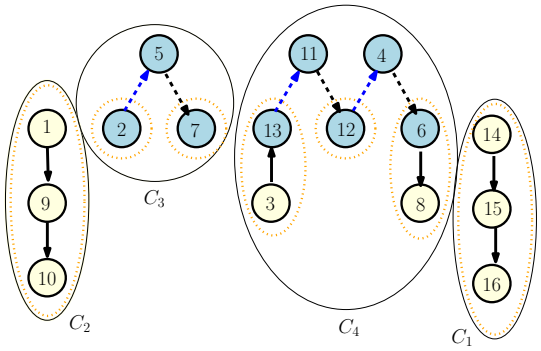

Fig. 3: A pseudochain partition found in example VI.1

if there is a feasible track assignment which assigns b to two different tracks. For short, we say that a triple (a, b, c) with the properties above is splittable.

In order to be feasible, some cars of $S(b)$ must then form a block at the end of one track containing the cars of $S(a)$ and a block at the beginning of some other track containig the cars of $S(c)$. It is easy to show the following Theorem:

Theorem V.1. *(Splittable Destinations) Given an instance $S \in \mathbb{S}^n$ of TMP. Let (a, b, c) be a triple of destinations with $I_a \cap I_b \neq \emptyset$ and $I_b \cap I_c \neq \emptyset$. Then the triple (a, b, c) is splittable if and only if there is no car of destination b between $first(c)$ and $last(a)$.*

Proof. Given a feasible track assignment and a destination b which is assigned to two tracks. We may assume that both tracks contain cars of other destinations. Let a be the destination preceding cars of b on the first track and c the destination following the cars on the other track. Then it is easy to see, that in the incoming train either $last(a) < first(c)$ or no car of destination b occurred between $first(c)$ and $last(a)$.

Conversely, let (a, b, c) a triple of destinations (a, b, c) as above. We start by assigning $S(a)$ to an open track, track 1 say, and cars of $S(c)$ to track 2. Cars of destination b will also be assigned to track 2 if they occur before $last(a)$, and to track one track 1 otherwise. All other cars are assigned to a destination-specific track. By assumption, either $last(a) < first(c)$, and the assignment is feasible. In the other case, after the first car of $S(c)$ is assigned to track 2, all remaining cars of $S(b)$ are assigned to the end of track 1. So in both cases the assignment is feasible. □

Observe that splittable triples cannot be read off from the comparability graph D itself. So in the next section we will enhance D to capture this extra information.

VI. MINPC AND TMP ARE EQUIVALENT

Let D be the comparability graph of the intervals given by an instance $S \in \mathbb{S}^n$ and (a, b, c), a splittable triple. Observe that by definition I_b overlaps both I_a and I_c. So $(a, b), (b, c) \notin A$. Let $B = \{(a, b), (b, c) \mid (a, b, c)$ is a splittable triple$\}$ and $D^* = D^*(S) = (V, A \cup B)$ be the *extended comparability graph* of S.

Example VI.1. *Given an instance $S \in \mathbb{S}^{50}$ with 16 destinations and*

$$\sigma = (1, 1, 2, 1, 2, 3, 3, 3, 4, 2, 2, 1, 5, 3, 3, 4, 2, 1, 1, 6, 6, 2, 5, 7, 8, 1, 9, 10, 8, 11, 12, 13, 2, 5, 8, 10, 14, 14, 15, 16, 16, 12, 7, 4, 10, 5, 7, 8, 13, 11)$$

A partition P of the extended comparability graph $D^*(S)$ in pseudochains is given by

- $\mathcal{C}_1 = \{14, 15, 16\}$ *with* $\ell(\mathcal{C}_4) = 1$.
- $\mathcal{C}_2 = \{1, 9, 10\}$ *with* $\ell(\mathcal{C}_2) = 1$.
- $\mathcal{C}_3 = \{2, 5, 7\}$ *with* $C_1 = \{2\}$, $b_2 = 5$, $C_2 = \{7\}$ *and* $\ell(P_1) = 2$.
- $\mathcal{C}_4 = \{3, 4, 6, 8, 11, 12, 13\}$ *with* $C_1 = \{3, 13\}$, $b_2 = 11$, $C_2 = \{12\}$, $b_3 = 4$, $C_3 = \{6, 8\}$ *and* $\ell(P_3) = 3$.

See Figure 3. The weight is $\ell(P) = 7$.

Lemma VI.2. *Let $S \in \mathbb{S}^n$ and P, a pseudochain partition of the extended comparability graph $D^*(S)$. Then P induces a feasible track assignment using $\ell(P)$ tracks.*

Proof. It suffices to show that we can assign a pseudochain $\mathcal{C} = C_1, b_2, C_2, b_3, C_3, ..., b_k, C_k$ of length k to k tracks. Let chain C_i begin with c_i and end with a_i. Let $B'_i = \{c \in b_i : c < last(c_{i-1})\}$ and $B''_i = \{c \in b_i : c > last(c_{i-1})\}$. We claim that, for $1 \leq i \leq k-1$, we can schedule the pseudochain such that track i contains B'_{i-1} followed by C_i again followed by B''_i. Suppose this is true for some $1 \leq j < k$. Since, by definition, the triple (a_j, b_{j+1}, c_{j+1}) is splittable, we may fill track j with B''_{j+1}, open track $j+1$ with B'_j and fill it with C_{j+1}. □

Lemma VI.3. *Let $S \in \mathbb{S}^n$ and tr, a feasible track assignment using k trains. Then tr induces a pseudochain partition P of the extended comparability graph with $\ell(P) = k$.*

Proof. Since tr is feasible, the cars of a destination d are assigned to at most two tracks and form a consecutive subsequence on their tracks. If they are assigned to two tracks they must be placed at the end of one track and at the beginning of some other track. Define a directed graph H on the set of destinations. Two destinations i, j are linked by an edge (i, j) if cars of i are placed immediately before cars of j on the same track. Then each connected component of H induces a pseudochain C of $D(S)$. Since $\ell(C)$ corresponds to the number of tracks used by the component, the claim follows. □

Example VI.4. *Consider the instance of example VI.1. The pseudochain partition P induces the following track assignment:*

Track 1 : $2_3, 2_{33}\ 5_{34}, 5_{51}$
Track 2 : $5_1, 5_{23}\ 7_{24}, 7_{47}$
Track 3 : $1_1, 1_{26}\ 9_{27}\ 10_{28}, 10_{45}$
Track 4 : $3_6, 3_{15}\ 13_{32}, 13_{49}\ 11_{50}, 11_{51}$
Track 5 : $11_1, 11_{30}\ 12_{31}, 12_{42}\ 4_{44}, 4_{51}$
Track 6 : $4_9, 4_{16}\ 6_{20}, 6_{21}\ 8_{25}, 8_{48}$
Track 7 : $14_{37}, 14_{38}\ 15_{39}\ 16_{40}, 16_{41}$

Here, the lower indices represent the position of the car in the input sequence. It is a feasible track assignment using $\ell(P) = 7$ tracks.

Theorem VI.5. *Let $S \in \mathbb{S}^n$ and $D^*(S)$, the extended comparability graph. Then minPC on $D^*(S)$ is equivalent to TMP.*

Proof. Follows from Lemma VI.2 and VI.3. □

Thus for every optimal solution of an instance $S \in \mathbb{S}^n$ of the TMP using $K(S)$ tracks, there exists a corresponding partition of the extended comparability graph $D(S)$ into pseudochains.

VII. A new Greedy-Approach: Greedy-PC

This greedy approach is based on the above observations on pseudochain partitions. Let $S \in \mathbb{S}^n$ be an instance of TMP and $T = (t_1, ..., t_{t_S})$ be the list of all splittable destination in $\mathcal{S}$ sorted increasingly by their left boundary. Our approach will return a partition of $D^*(S)$ into pseudochains.

Given a splittable destination $(a, b, c) \in T$, the function add applied to a pseudochain P returns `true` if the triple can be added to P and `false` otherwise. We follow the idea to have the best solution within this chain P and try to add every possible splittable triple in T to a pseudochain. We will redo this as long as nodes remain in $\mathcal{S}$.

The worst-case runtime of this heuristic is $f(n) = (\frac{1}{2}n^3 + n^2) = O(n^3)$. See algorithm 1 for an implementation in pseudocode.

VIII. Experimental Results

We used Python 3.4 with NetworkX for creating random instances and implement the greedy heuristic as well as the Linear Programming relaxation introduced by Beygang [9]. Four 2.4 GHz processors and 8 GB RAM were available running Linux Kernel 3.10. We used GLPK (GNU Linear Programming Kit) 4.52 to solve the linear program. To get comparable results, we followed [9] to create random instances. This function takes the number n of cars and computes uniform and independent problem instances.

The greedy heuristic introduced by Beygang et al. ([9] and [1]) leads to the upper bound denoted by *Coloring*. It is equivalent to a graph coloring approach for the interval graph $G(S)$. The runtime is $O(n^2)$. Algorithm 1 has also polynomial runtime in $O(n^3)$.

We approximate the optimal solution according to the bounds u_{lp} and l_{lp}, the upper and lower bound given by the solution of the linear program introduced by [9]. It was observed in [12] that the lower bound very often coincides with the value of the optimal solution.

Figures 4 and 5 summarize the output of the heuristics on 50 random instances with a fixed number of cars. For a small number of cars the distance between *Coloring* and u_{greedy} is small, but notable, see figure 4. We notice the greedy approach can lead to solutions using more tracks than the *Coloring* approach. The situation changes significantly for instances with more cars. Figure 5 shows that Greedy-PC performs better than *Coloring* on instances with 300 cars.

Algorithm 1 Greedy-PC

Require: Extended comparability graph $D^*(S)$ with its maximal stable set $\mathcal{S}$ in $D(S)$ and a list $T = (t_1, ..., t_{t_S})$ of splittable destinations in $\mathcal{S}$.
Ensure: Partition of $D^*(S)$ in pseudochains

```
visited = ∅
count = 0
while |T| > 0 do
    count++
    P.add(pseudochain P_count)
    for every (a, b, c) = t_i ∈ T do
        if P_count.add(a, b, c) = true then
            visited.add a, b, c
        end if
    end for
    for every v ∈ visited do
        delete every t_i containing v from T
    end for
end while
for every node v ∈ V(G) do
    if v ∉ visited then
        for i = 1, ..., count do
            if P_i.add(v) = true then
                visited.add v
                exit
            end if
        end for
        count++
        P.add(pseudochain P_count)
        P_count.add(v)
    end if
end for
return P
```

IX. Conclusions

We have introduced and discussed pseudochain partitions and their relation to the Train Marshalling Problem. There is only little discussion about TMP in the literature, but the problem has an intimate relationship to other sorting problems of rolling stock, see [4]. Thus it is an important step to provide a better understanding of the underlying graph structures. Pseudochain partitions directly lead to a new heuristic providing a improved upper bounds for optimal solutions of TMP. We could proof that every optimal solution of TMP is equivalent to a minimal partition of the corresponding extended comparability graph $D^*(S)$ into pseudochains.

The greedy approach has been evaluated for 2 instances with 100 and 300 cars, each consisting of 50 random instances each. The computational results show that the model is useful and the proposed Greedy-approach performs in general significantly better than other state-of-the art approaches.

To sum up, although we achieved encouraging results, there are still questions which are not answered or even discussed in this paper. For example, can the inherent structure of

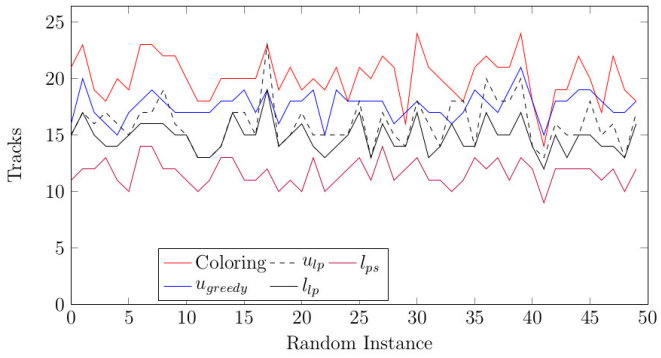

Fig. 4: Results for random instances with $n = 100$ cars. *Colouring* is a Greedy-approach introduced by Beygang, u_{lp} and l_{lp} are upper and lower bounds of the integer linear program approach, see [9]. The lower bound l_{ps} was introduced in [12]. u_{greedy} shows the results of our novel algorithm 1.

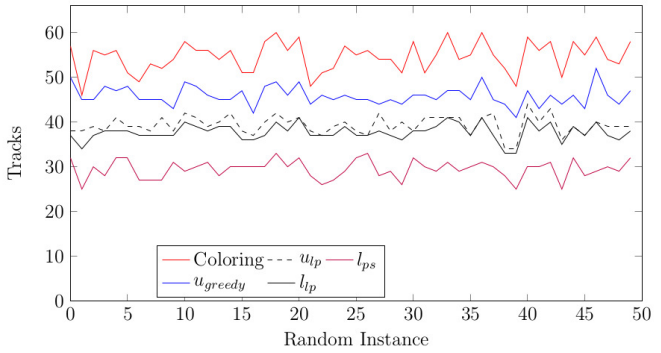

Fig. 5: Results for random instances with $n = 300$ *Colouring* is a Greedy-approach introduced by Beygang, u_{lp} and l_{lp} are upper and lower bounds of the integer linear program approach, see [9]. The lower bound l_{ps} was introduced in [12]. u_{greedy} shows the results of our novel algorithm 1.

pseudoschains be used to find even better heuristics than those discussed in this paper? Are there any instances of the TMP that can be solved in polynomial time?

The results encourages the further improvement on heuristics to solve minPC and the application of this method to other sorting of rolling Stock Problems.

REFERENCES

[1] K. Beygang, F. Dahms, and S. O. Krumke, "Train marshalling problem - algorithms and bounds," University of Kaiserslautern, Tech. Rep. 132, 2010.

[2] W. Hiller, *Rangierbahnhöfe*. Berlin: VEB Verlag für Verkehrswesen, 1983.

[3] M. Giger, "Rangierbahnhof Limmattal," *SWISS ENGINEERING STZ AUTOMATE NOW!*, vol. 1-2, pp. 93–94, 2010.

[4] R. S. Hansmann, *Optimal Sorting of Rolling Stock*. Göttingen: Cuvillier, 2011.

[5] Y. Zhu and R. Zhu, "Sequence reconstruction under some order-type constraints," *Scientia Sinica (A)*, vol. 26(7), pp. 702–713, 1983.

[6] E. Dahlhaus, P. Horak, M. Miller, and J. Ryan, "The train marshalling problem," *Discrete Applied Mathematics*, vol. 103(1-3), pp. 41–54, 2000. [Online]. Available: https://doi.org/10.1016/s0166-218x(99)00219-x

[7] L. Brueggeman, M. Fellows, R. Fleischer, M. Lackner, C. Komusiewicz, Y. Koutis, A. Pfandler, and F. Rosamond, "Train marshalling is fixed parameter tractable," in *Fun with Algorithms*, E. Kranakis, D. Krizanc, and F. Luccio, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 51–56. [Online]. Available: https://doi.org/10.1007%2F978-3-642-30347-0_8

[8] E. Dahlhaus, F. Manne, M. Miller, and J. Ryan, "Algorithms for combinatorial problems related to train marshalling," *Proceedings of AWOCA 2000, Hunter Valley*, pp. 7–16, 2000.

[9] K. Beygang, *On the Solution of Some Railway Freight Car optimization Problems*. München: Dr. Hut, 2011.

[10] F. Rinaldi and R. Rizzi, "Solving the train marshalling problem by inclusion–exclusion," *Discrete Applied Mathematics*, vol. 217, Part 3, pp. 685 – 690, 2017.

[11] J. T. Haahr and R. M. Lusby, "A matheuristic approach to integrate humping and pullout sequencing operations at railroad hump yards," *Networks*, vol. 67, no. 2, pp. 126–138, 2016.

[12] J. Dörpinghaus, *Pseudostabile Mengen in Graphen*. Fraunhofer Verlag, 2018. [Online]. Available: https://kups.ub.uni-koeln.de/8066/