# Università degli studi di Udine

## Solving the train marshalling problem by inclusion-exclusion

(Article begins on next page)

25 April 2024

# Solving the train marshalling problem by inclusion–exclusion

Franca Rinaldi and Romeo Rizzi

**Abstract**

In the Train Marshalling Problem (TMP) the cars of a train having different destinations have to be reordered in such a way that all the cars with the same destination appear consecutively. To this aim the cars are first shunted on $k$ auxiliary rails, then the sequences of cars present on the different rails are reconnected one after each other to form a new train. The TMP is the problem of minimizing the number $k$ of auxiliary rails needed to obtain a train with the required property. The TMP is an NP-hard problem. Here we present an exact dynamic programming algorithm for the TMP based on the inclusion–exclusion principle. The algorithm has polynomial space complexity and time complexity that is polynomial in the number of cars, exponential in the number of destinations. This shows that the TMP is fixed parameter tractable with the number of destinations as parameter.

**Keywords**: train marshalling, fixed parameter tractability, inclusion-exclusion principle, dynamic programming.

# 1  Introduction

In the Train Marshalling Problem (TMP) a train $T = (a_1, \ a_2, \ \dots \ , a_n)$ with $n$ cars having $t$ different destinations is given. The order of the cars of the train can be modified by means of $k$ auxiliary rails (or tracks) as follows. First the $n$ cars $a_1, a_2, \dots, a_n$ are considered one by one in their initial order and each car is moved to one of the $k$ auxiliary rails and placed behind the cars already on the rail. This process creates $k$ sequences of cars, one for each rail. Then a new train is created by reconnecting these sequences in a single one by sequencing the cars on the first rail in their actual order followed by the cars on the second rail and so on, ending with the cars on the $k$th rail. The TMP is the problem of finding the minimum number $k$ of auxiliary rails needed to rearrange the train in such a way that all the cars with the same destination appear consecutively. An order of the cars that satisfies this condition will be called a TM-order. The decision version of the problem (DTMP) is the problem of deciding if, given a natural number $k$, a TM-order of a train $T$ can be obtained by using at most $k$ auxiliary rails.

The TMP was originally proposed in [15] by Zhu and Zhu who studied some polynomial classes of the problem. In [7] Dalhaus et al. showed that the DTMP is NP-complete and, as a consequence, the TMP is an NP-hard problem. They also proved that the optimal value of the TMP is upperbounded by the value

$$L(n,t) = \min\{t, \lceil n/4 + 1/2 \rceil\}. \tag{1}$$

A 2-approximation algorithm for the TMP based on an interval graph coloring approach has been proposed by Dahlhaus et al. in [8]. The question if the TMP is a fixed parameter tractable problem has been addressed by Brueggemann et al. in [6]. In this paper the authors describe a dynamic programming procedure for the DTMP that requires time $O(2^{O(k)} poly(n))$ and space $O(n^2 k 2^{8k})$, thus polynomial for each fixed value of the parameter $k$. However, the proof of correctness of the algorithm contains a bug which, to the authors' knowledge, has not been fixed yet [9].

In this paper we present an exact algorithm that solves a graph theoretical model of the DTMP. The algorithm is a dynamic programming procedure based on the principle of inclusion-exclusion and has time complexity $O(nkt^2 2^t)$ and polynomial space complexity $O(nkt)$. This procedure can be easily adopted to solve the TMP by binary search in $O(nt^2 2^t L \log_2 L)$ where $L = L(n,t)$ is the upperbound given in (1). In particular, this shows that the TMP is fixed parameter tractable with respect the number $t$ of different destinations. Because of the exponential factor $2^t$ in the time complexity, the procedure can be reasonably used to solve instances with a number of destinations $t$ up to 30. It remains an open problem if the DTMP shows the stronger property to be fixed parameter tractable even with respect to the number $k$ of auxiliary rails.

The TMP belongs to a wide class of combinatorial problems which arise in the optimization of the train classification processes. These problems usually require to rearrange the cars of a train (or of a set of trains) to sequence them in an assigned order. So, differently from the TMP, the final order of the cars is usually an input of the problem. The goal is reached by partitioning the cars of the train on a given set of auxiliary tracks and then performing a sequence of so called *roll-in* operations, where each roll-in operation takes the cars on a track and suitably routes them on the other tracks. The main objective is either the minimization of the number of roll-in operations or the number of cars globally involved in these operations. Furthermore, scheduling aspects can also affect the problem. For a detailed description of the train classification problems see for instance [10, 11] and the references therein.

We remind that the inclusion-exclusion principle has been successfully applied in combinatorial optimization to solve classical NP-problems, in particular graph theory problems. Initially, Karp [12] used this approach to solve the hamiltonian path problem, the bin packing problem and some sequencing problems. Later on, other combinatorial problems have been addressed by the inclusion-exclusion approach in [1, 2, 3, 4]. In particular, Björklund et al. [5] have proposed a general way to solve a class of set partitioning problems including chromatic number, domating number, maximum $k$-cut and list coloring.

The remainder of the paper is organized as follows. In Section 2 we formally define the TMP. In Section 3 we introduce a graph theoretical model for the DTMP suitable to be solved by the inclusion-exclusion principle. In Section 4 we present two dynamic programming procedures that solve the DTMP and the TMP, respectively. Some conclusions are drawn in Section 5.

**Notation** We will use the following notation. For each $n \in \mathbb{N}$, $\mathbb{N}_n = \{1, \ldots, n\}$ denotes the set of integers in between 1 and $n$ and for $i, j \in \mathbb{N}$, $i < j$, we denote by $[i, j]$ the set of integers $r$ with $i \leq r \leq j$. Given a sequence $\alpha$ of length $n$ and a subset $S \subset \mathbb{N}_n$ we denote by $\alpha[S]$ the sequence obtained from $\alpha$ by removing the elements in positions in $\mathbb{N}_n \setminus S$. If $S = \{i\}$, we also write $\alpha[S] = \alpha[i] = \alpha_i$. Given two sequences $\alpha$ and $\beta$ of length $m$ and $n$, respectively, we denote by $\alpha \cdot \beta$ the concatenation of $\alpha$ and $\beta$, that is the sequence $(\alpha_1, \ldots, \alpha_m, \beta_1, \ldots, \beta_n)$. Finally, any (total) order of a set $\mathbb{N}_n$ is represented by the sequence $\tau$ of length $n$ where $\tau[i]$, $i = 1, \ldots, n$, denotes the $i$-th element in the order.

# 2 Problem formulation

An instance of the TMP consists in a triple $(n, t, \mathcal{D})$ where $n$ is the number of cars of the train, $t$ is the number of destinations and $\mathcal{D}$ is a partition of $\mathbb{N}_n$ in subsets $D(j)$, $j \in \mathbb{N}_t$. Each set $D(j)$ contains the indices of the cars having destination $j$. For the sake of simplicity, in the following we identify the cars of the train with their index. In this way the original order of the cars corresponds to the sequence $(1, 2, \ldots, n)$.

**Definition 1.** *An order $\tau$ of $\mathbb{N}_n$ is said a TM-order for the instance $(n, t, \mathcal{D})$ if the elements of each set $D(j)$, $j \in \mathbb{N}_t$, appear consecutively in $\tau$, i.e., $\tau[r], \tau[s] \in D(j)$ for some $1 \leq r < s \leq n$ implies $\tau[i] \in D(j)$ for every $r \leq i \leq s$.*

To formally define the TMP we observe that the train $\alpha = (1, 2, \ldots, n)$ can be reordered by means of $k$ auxiliary tracks to obtain a TM-train if and only if there exists a map $\phi : \mathbb{N}_n \to \mathbb{N}_k$ such that, setting $\phi^{-1}(r) = \{i \in \mathbb{N}_n : \phi(i) = r\}$ for each $r \in \mathbb{N}_k$, the sequence

$$\tau^\phi = \alpha[\phi^{-1}(1)] \cdot \alpha[\phi^{-1}(2)] \cdots \alpha[\phi^{-1}(k)]$$

is a TM-order of $\mathbb{N}_n$. In this case the map $\phi$ is said a $k$-TM-solution, or briefly a $k$-solution, of the TMP instance. Each set $\phi^{-1}(r)$ denotes the cars that are moved on the $r$-th track. According to $\alpha[\phi^{-1}(r)]$, these cars are sequenced on track $r$ for increasing indices. Let us denote by $\pi^\phi$ the order of $N_t$ in which the $t$ destinations appear in $\tau^\phi$, i.e., such that for each $j \in \mathbb{N}_t$ the cars in $D(\pi^\phi(j))$ appear in $\tau^\phi$ just after the cars in $D(\pi^\phi(j-1))$.

The TMP and its decision version DTMP can be formulated as follows.

**Train Marshalling Problem**: Given a TMP instance $(n, t, \mathcal{D})$ find the minimum $k \in \mathbb{N}$ such that there exists a $k$-solution.

**Decision Train Marshalling Problem (DTMP)** Given a TMP instance $(n, t, \mathcal{D})$ and $k \in \mathbb{N}$, determine if there exists a $k$-solution.

**Example 1**. Consider the TMP instance defined by $n = 10$, $t = 5$ and

$$D_1 = \{1, 8\}, \quad D_2 = \{2, 9\}, \quad D_3 = \{3, 7\}, \quad D_4 = \{4, 6\}, \quad D_5 = \{5, 10\}.$$

The map $\phi : \mathbb{N}_{10} \to \mathbb{N}_3$ defined by $\phi(1) = \phi(8) = \phi(9) = 1$, $\phi(2) = \phi(4) = \phi(6) = \phi(7) = 2$, $\phi(3) = \phi(5) = \phi(10) = 3$ is a 3-solution since the order $\tau^\phi = (1, 8, 9, 2, 4, 6, 7, 3, 5, 10)$ is a TM-order. The order $\pi^\phi$ induced on the destination-set $\mathbb{N}_5$ is $\pi^\phi = (1, 2, 4, 3, 5)$.

The TMP has been alternatively formulated by Donald Knuth [13] as follows. Given a TMP instance $(n, t, \mathcal{D})$ and $k \in \mathbb{N}$, denote by $\sigma(n, k)$ the sequence

$$\sigma(n, k) = (\underbrace{1, 2, \ldots, n}_{1}, \underbrace{1, 2, \ldots, n}_{2}, \ldots, \underbrace{1, 2, \ldots, n}_{k}) \tag{2}$$

obtained by replicating $k$ times the sequence $(1, 2, \ldots, n)$.

**Definition 2.** *We say that the sequence $\sigma(n, k)$ covers the partition $\mathcal{D}$ if there exists a subset $R \subseteq \mathbb{N}_{nk}$, $|R| = n$, with the property that $\sigma(n, k)[R]$ is a TM-order of $\mathbb{N}_n$. In this case we say that $\sigma(n, k)$ covers $\mathcal{D}$ according to R.*

It is easy to verify that the sequence $\sigma = \sigma(n, k)$ covers $\mathcal{D}$ if and only if the TMP instance admits a $k$-solution. Assume first that $\sigma$ covers $\mathcal{D}$ according to $R \subseteq \mathbb{N}_{nk}$ and write the TM-order $\sigma[R]$ as

$$\sigma[R] = \sigma[R \cap [1, \ldots, n]] \cdots \sigma[R \cap [n(r-1)+1, \ldots, nr]] \cdots \sigma[R \cap [n(k-1)+1, \ldots, nk]].$$

Let $\phi$ be the application that maps into $r$, $r \in \mathbb{N}_k$, all the elements that appear in $\sigma[R \cap [n(r-1)+1, \ldots, nr]]$. Then $\phi$ induces the order $\tau^\phi = \sigma[R]$ and thus it is a $k$-solution of the TMP. Conversely, given a $k$-solution $\phi$, the sequence $\sigma(n, k)$ covers $\mathcal{D}$ according to the subset $R = \cup_{r=1}^k \{i \in [n(r-1)+1, nr] : \phi(i) = r\}$.

**Example 1 (continue).** Consider again Example 1. The sequence $\sigma(10, 3)$ covers $\mathcal{D}$ according to the set $R = \{1, 8, 9, 12, 14, 16, 17, 23, 25, 30\}$ which leads to the TM-permutation

$$\sigma(10, 3)[R] = (1, 8, 9, 2, 4, 6, 7, 3, 5, 10) = \tau^\phi.$$

# 3  A graph model for the DTMP

Given a TMP instance $(n, t, \mathcal{D})$ and $k \in \mathbb{N}$, define a directed graph $G(n, t, \mathcal{D}, k) = (V, E)$ as follows. The node-set $V = \{1, 2, \ldots, nk\} \cup \{0, nk + 1\}$ contains a node for each position in the sequence $\sigma = \sigma(n, k)$ and two auxiliary nodes $0$ and $nk + 1$. In order to define the arc-set $E$, consider the function $\psi$

$$\psi : \{0, 1, \ldots, nk\} \times \{1, \ldots, t\} \to \{1, \ldots, nk\}$$

defined by

$$\psi(i, j) = \min\{\min\{l > i : D(j) \subseteq [i+1, l]\}, nk + 1\}. \tag{3}$$

**Algorithm 1** GRAPH(): constructs the graph $G(n, t, \mathcal{D}, k)$ associated to a DTMP instance.

**Input:** a DTMP instance $(n, t, \mathcal{D}, k)$;
**Output:** the graph $G(n, t, \mathcal{D}, k)$;

**Initialization:** $V = \mathbb{N}_{nk+1}$; $E = \emptyset$;

**for** $i = 0$ to $nk - 1$ **do**
  **for** $j = 1$ to $t$ **do**
    $s_j := 0$;
  **end for**
  **for** $h = i + 1$ to $\min\{i + n, nk\}$ **do**
    let $\sigma(n, k)[h] \in \mathcal{D}(j)$;
    $s_j := s_j + 1$;
    **if** $s_j = |D(j)|$ **then**
      $E := E \cup (i, h)$; $C[(i, h)] := j$;
    **end if**
  **end for**
**end for**
**for** $i = 1$ to $nk$ **do**
  $E := E \cup (i, nk + 1)$; $C[(i, nk + 1)] := t + 1$;
**end for**
**return**    $G := (V, E)$;

For each pair $i, j$ of indices the function $\psi$ returns the minimum index $l$ such that all the elements of $D(j)$ appear in $\sigma$ from position $i + 1$ to position $l$ if such position exists, it returns $nk + 1$ otherwise. Clearly, whenever $\psi(i, j) < nk + 1$, it holds $\sigma[\psi(i, j)] \in D(j)$.

The graph $G(n, t, \mathcal{D}, k)$ has arc-set $E = \hat{E} \cup E_{nk+1}$ where

$$\hat{E} = \{(i, \psi(i, j)) \mid i \in \{0, 1, \ldots, nk\},\ j \in \mathbb{N}_t,\ \psi(i, j) \neq nk + 1\} \tag{4}$$

and

$$E_{nk+1} = \{(i, nk + 1) \mid i \in V \setminus \{nk + 1\}\}.$$

We now color the arcs of the graph $G(n, t, \mathcal{D}, k)$ with $t + 1$ different colors, by assigning color $j$ to every arc $(i, h) \in \hat{E}$ such that $\sigma[h] \in D(j)$ and by assigning color $t + 1$ to each arc of $E_{nk+1}$. In the following we denote by $C[e]$ the color of the arc $e \in E$ and we call $j$-arc any arc of color $j$.

**Definition 3.** *Let $J \subseteq \mathbb{N}_t$. A $J$-rainbow path in the graph $G(n, t, \mathcal{D}, k)$ is a directed path $P$ with $|J|$ arcs that contains (exactly) one arc of color $j$ for each $j \in J$.*

5

The next result links the yes/no answer for the DTMP to the existence of $\mathbb{N}_{t+1}$-rainbow paths in the graph $G = (n, t, \mathcal{D}, k)$.

**Theorem 1.** *A TMP instance $(n, t, \mathcal{D})$ admits a $k$-solution if and only if the directed graph $G(n, t, \mathcal{D}, k)$ contains an $\mathbb{N}_{t+1}$-rainbow path from node $0$ to node $nk + 1$.*

*Proof* Assume that $\sigma = \sigma(n, k)$ covers $\mathcal{D}$ according to the set $R \subseteq \mathbb{N}_{nk}$ and let $\pi$ be the order of $\mathbb{N}_t$ in which the destinations appear in the sequence $\sigma[R]$. Define $h(0) = 0$ and, according to (3), $h(j) = \psi(h(j-1), \pi(j))$ for each $j \in \mathbb{N}_t$, so that $h(j)$ is either the minimum index in $\mathbb{N}_{nk}$ such that the sequence $\sigma[h(j-1)+1, \ldots, h(j)]$ contains all the elements of the destination-set $D(\pi(j))$ or $nk+1$. Since it clearly holds $h(j) \leq \max\{i \in R : \sigma[i] \in D(\pi(j))\}$ we get $h(j) \leq nk$ for each $j$. Then each pair $(h(j-1), h(j))$ is an arc of $\hat{E}$ of color $\pi(j)$ and the path $P$ that visits in sequence the nodes $0, h(1), \ldots, h(t)$ and $nk+1$ is an $\mathbb{N}_{t+1}$-rainbow path from $0$ to $nk+1$ in the graph $G = (n, t, \mathcal{D}, k)$. Conversely, let $(0 = h(0), h(1), \ldots, h(t), h(t+1) = nk+1)$ be the sequence of nodes visited by an $\mathbb{N}_{t+1}$-rainbow path $P$ in the graph $G(n, t, \mathcal{D}, k)$. Define an order $\pi$ of $\mathbb{N}_t$ by setting $\pi(j) = C[(h(j-1), h(j))]$, i.e., the color of the $j$-th arc of $P$. Then $h(j) = \psi(h(j-1), \pi(j)) > h(j-1)$ and $D(\pi(j)) \subseteq \{\sigma[r] : r \in [h(j-1)+1, h(j)]\}$ for each $j \in \mathbb{N}_t$. So if we define $R(j) = \{r \in [h(j-1)+1, h(j)] : \sigma[r] \in D(\pi(j))\}$, $j \in \mathbb{N}_t$, and $R = \cup_{j=1}^{t} R(j)$ we obtain that the sequence $\sigma[R]$ is a TM-order of $\mathbb{N}_n$. $\square$

**Example 2**. Consider the DTMP instance defined by $n = 4$, $t = 3$, $D(1) = \{1, 3\}$, $D(2) = \{2\}$, $D(3) = \{4\}$ and $k = 2$. The associated graph $G(4, 3, \mathcal{D}, 2)$ is shown in Figure 1 where dotted, plain, dashed and gray arcs correspond to arcs of color 1, 2, 3 and $t+1 = 4$, respectively. Since the graph contains the $\mathbb{N}_4$-rainbow path visiting the sequence of nodes $(0, 2, 5, 8, 9)$ the instance admits a 2-solution. The corresponding order of $\mathbb{N}_3$ is $\pi = (2, 1, 3)$ and the sequence $\sigma(n, 2)$ covers $\mathcal{D}$ according to the set $R = R(1) \cup R(2) \cup R(3) = \{2\} \cup \{3, 5\} \cup \{8\}$. The final $TM$-order $\tau = (2, 3, 1, 4)$ is obtained by mapping cars 2 and 3 on the first track and cars 1 and 4 on the second track.

Algorithm 1 reports the procedure GRAPH() that, given a TMP instance $(n, t, \mathcal{D})$ and $k \in \mathbb{N}$, constructs the graph $G(n, t, \mathcal{D}, k)$. For each position $i = 1, \ldots, nk - 1$ the procedure scans the elements of $\sigma(n, k)$ from position $i + 1$ to position $i + n$ by keeping track of their membership to the sets $D(j)$. When the $|D(j)|$-th occurrence of an element in $D(j)$ is encountered in position $h$, then an arc $(i, h)$ of color $j$ is added to the arc-set. Finally, the procedure adds all the arcs $(i, nk + 1)$, $i \in V \setminus \{nk + 1\}$, of color $t + 1$. The procedure requires time $O(n^2 k)$. Moreover, since each node of the graph $G(n, t, \mathcal{D}, k)$ has at most $t + 1$ exiting arcs, the space needed to store the graph $G(n, t, \mathcal{D}, k)$ is $O(nkt)$.
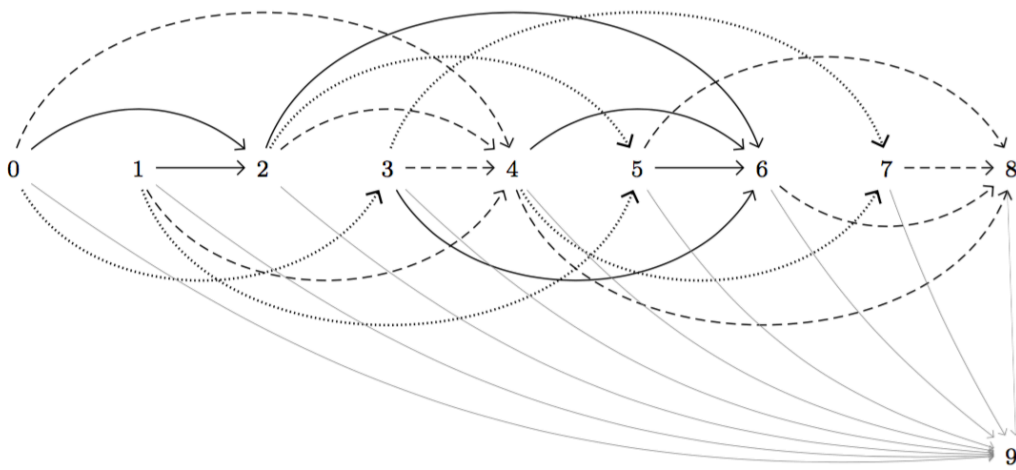
Figure 1: The graph $G(n, t, \mathcal{D}, k)$ representing the instance in Example 2. The graph is as explicitly constructed by Algorithm 1. As for the "colors" of the arcs: dotted, plain, dashed arcs correspond to destinations 1, 2 and 3, respectively, while the gray arcs are the additional arcs of the set $E_{nk+1}$.

# 4 Exact algorithms for the DTMP and the TMP

Let $G(n, t, \mathcal{D}, k)$ be the graph corresponding to the DMPT instance $(n, t, \mathcal{D}, k)$. The existence of an $\mathbb{N}_{t+1}$-rainbow path in the graph $G(n, t, \mathcal{D}, k)$ can be established by means of an inclusion-exclusion argument. The inclusion-exclusion principle can be stated as follows ([14]).

**Theorem 2.** *Let $U$ be a finite set and $P_1, \ldots, P_t$ subsets of $U$. Then*

$$|P_1 \cap \cdots \cap P_t| = \sum_{T \subseteq \mathbb{N}_t} (-1)^{|T|} |\cap_{j \in T} \bar{P}_j| \tag{5}$$

*where $\bar{P}_j = U \setminus P_j$ and, by convention, $\cap_{j \in \emptyset} \bar{P}_j = U$.*

In the case of the DTMP, let us define $U$ as the set of all the paths in $G(n, t, \mathcal{D}, k)$ from node 0 to node $nk + 1$ with $t + 1$ arcs and $P_j$, $j = 1, \ldots, t$, as the subset of the paths of $U$ that contain at least one arc of color $j$. Then the intersection $P_1 \cap \cdots \cap P_t$ is just the set of all the $\mathbb{N}_{t+1}$-rainbow paths from 0 to $nk + 1$. Applying the inclusion exclusion formula (5) we obtain that the number $X$ of the $\mathbb{N}_{t+1}$-rainbow paths from 0 to $nk + 1$ in $G(n, t, \mathcal{D}, k)$ is given by

$$X = \sum_{T \subseteq \mathbb{N}_t} (-1)^{|T|} N(T) \tag{6}$$

where, for each subset $T \subseteq \mathbb{N}_t$, $N(T)$ denotes the number of paths in $U$ that do not contain arcs with color in $T$.

Now, for each subset $T \subseteq \mathbb{N}_t$, the value $N(T)$ can be computed by dynamic programming as follows. For each length $l = 1, \ldots, t + 1$ and each node $i = 0, \ldots, nk$ of $V$ define $N_T(i, l)$ as the number of paths of length $l$ from node $i$ to node $nk + 1$ whose arcs have color in $\mathbb{N}_t \setminus T$, so that $N(T) = N_T(0, t + 1)$. The values $N_T(i, l)$ can be computed by means of the recursion

$$N_T(i, l) = \sum_{(i,h) \in E : C[(i,h)] \notin T} N_T(h, l - 1)$$

and the base conditions $N_T(i, 1) = 1$ for each $i \in \mathbb{N}_{nk}$. The procedure DYN_PROG() just described is reported in Algorithm 2. It can be implemented by using two arrays of length $nk + 1$ to store the values $N_T(i, l)$ and $N_T(i, l - 1)$, thus it requires $O(nk)$ space. Since there are at most $t + 1$ arcs exiting from each node of the graph $G(n, t, \mathcal{D}, k)$, its time complexity is $O(nkt^2)$.

Algorithm 3 reports the scheme of the algorithm DTMP() that solves the DTMP. The algorithm first calls for the procedure GRAPH() to construct the graph $G(n, t, \mathcal{D}, k)$

**Algorithm 2** DYN_PROG(): given $G(n, t, \mathcal{D}, k)$ and a set $T \subseteq \mathbb{N}_t$ computes the number of paths from node 0 to node $nk + 1$ with $t + 1$ arcs of color in $\mathbb{N}_t \setminus T$.

---

**Input:** $G = G(n, t, \mathcal{D}, k)$, $T \subseteq \mathbb{N}_t$;
**Output:** $N(T)$;

**for** $i = 0$ to $nk$ **do**
$\quad N_T(i, 1) = 1$;
**end for**
**for** $l = 2$ to $t + 1$ **do**
$\quad$ **for** $i = 0$ to $nk$ **do**
$\quad\quad N_T(i, l) := 0$;
$\quad\quad$ **for all** $e = (i, h) \in \hat{E}$, $C[(i, h)] \notin T$ **do**
$\quad\quad\quad N_T(i, l) := N_T(i, l) + N_T(h, l - 1)$;
$\quad\quad$ **end for**
$\quad$ **end for**
**end for**
**return** $N_T(0, t + 1)$;

---

corresponding to the DTMP instance. Then it computes the number $X$ of $\mathbb{N}_{t+1}$-rainbow paths in the graph according to (6) by summing up the values $(-1)^{|T|} N(T)$ where, for each $T \subseteq \mathbb{N}_t$, $N(T)$ is obtained by calling procedure DYN_PROG(). The algorithm has an overall time complexity $O(nkt^2 2^t)$ which is exponential in the number $t$ of destinations. The space complexity $O(nkt)$ is determined by the space required to store the graph $G(n, t, \mathcal{D}, k)$.

The algorithm for the DMPT described above can be used to solve by binary search the optimization problem TMP. In order to limit the range of the search, one can use the fact that, as shown in [7], the optimal value of every TMP instance with $n$ cars is at most

$$L(n, t) = \min\{t, \lceil n/4 + 1/2 \rceil\}.$$

This bound can be further reduced for instances in which the train has consecutive cars with the same destination. Indeed, as it easy to show, the following result holds.

**Lemma 1.** *Let $(n, t, \mathcal{D})$ be a TMP instance with two consecutive cars $i$ and $i + 1$ having the same destination. Then the instance $(n - 1, t, \mathcal{D}')$ obtained by removing car $i$ has the same optimal value.*

We call *contracted instance* $C(n, t, \mathcal{D})$ of $(n, t, \mathcal{D})$ the instance obtained by recursively removing a car from the initial train if the following one has its same destination.

The following theorem summarizes the previous results.

9

**Algorithm 3** DTMP() : finds the number of $t+1$-rainbow paths in the graph $G(n,t,\mathcal{D},k)$ associated to a DTMP instance

---

**Input:** a DTMP instance $(n,t,\mathcal{D},k)$;
**Output** the number $X$ defined in (6);

G:= GRAPH$(n,t,\mathcal{D},k)$;
$X := 0$;
**for all** $T \subseteq \mathbb{N}_t$ **do**
   $N(T) := $ DYN_PROG$(G,\ T)$;
   $X := X + (-1)^{|T|}N(T)$;
**end for**
**return** $X$;

---

**Theorem 3.** *Every TMP instance $(n,t,\mathcal{D})$ can be solved by a procedure requiring $O(\bar{n}Lt)$ space and $O(\bar{n}t^2 2^t L \log_2 L)$ time where $L = \min\{t, \lceil \frac{\bar{n}}{4} + \frac{1}{2} \rceil\}$ and $\bar{n}$ is the number of cars in the contracted instance $C(n,t,\mathcal{D})$.*

# 5 Conclusions

The paper presents an exact algorithm for the Train Marshalling Problem having polynomial space complexity and time complexity which is polynomial in the number $n$ of cars of the train and exponential (according to a $2^t$ factor) in the number $t$ of destinations. This proves that the Train Marshalling Problem is fixed parameter tractable with the number $t$ of destinations as parameter.

# References

[1] E. T. Bax, Inclusion and exclusion algorithm for the Hamiltonian Path Problem, Information Processing Letters 47 (1993), 203-207.

[2] E. T. Bax, Algorithms to count paths and cycles, Informormation Processing Letters 52 (1994), 249-252.

[3] E. T. Bax, J. Franklin, A finite-difference sieve to count paths and cycles by length, Informormation Processing Letters 60 (1996), 171-176.

[4] A. Björklund, T. Husfeldt, Exact algorithms for exact satisfiability and number of perfect matchings, Algorithmica 52 (2008), 226-249.

[5] A. Björklund, T. Husfeldt, M. Koivisto, Set Partitioning via Inclusion-Exclusion, SIAM Journal on Computing 39 (2009), 546-563.

[6] L. Brueggeman, M. Fellows, R. Fleischer, C. Komusiewicz, Y. Koutis, F. Rosamond, Train Marshalling is fixed Parameter Tractable, in: E. Kranakis, D. Krizanc and F. Luccio (Eds.), FUN 2012, LNCS 7288, Springer, Berlin, 2012, 51-56.

[7] E. Dahlhaus, P. Horak, M. Miller, J. F. Ryan, The train marshalling problem, Discrete Applied Mathematics 103 (2000), 41-54.

[8] E. Dahlhaus, F. Manne, M. Miller, J. F. Ryan, Algorithms for Combinatorial Problems Related to Train Marshalling, in: Proceedings of AWOCA 2000, Hunter Valley, 2000, 7-16.

[9] Mike Fellows and Rudolf Fleischer, *private communication*.

[10] M. Gatto, J. Maue, M. Mihalák, P. Widmayer, Shunting for dummies: An introductory algorithmic survey in: Robust and Online Large-Scale Optimization, LNCS 5868, Springer, Berlin, 2009, 310-337.

[11] R. Jacob, P. Márton, J. Maue, M. Nunkesser, Multistage methods for freight train classification, Networks 57 (2011), 87-105.

[12] R. M. Karp, Dynamic Programming meets the Principle of Inclusion and Exclusion, Operations Research Letters 1 (1982), 49-51.

[13] D. Knuth, *personal letter*.

[14] H.J. Ryser, Combinatorial Mathematics, The Mathematical Association of America, John Wiley, New York, 1963.

[15] Y.Zhu, R.Zhu, Sequence reconstruction under some order-type constraints, Scientia Sinica Series A 26 (1983), 702-713.