Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT

Electronic Notes in Theoretical Computer Science

Electronic Notes in Theoretical Computer Science 92 (2004) 16–33
www.elsevier.com/locate/entcs

A Graph Theoretical Approach To The Shunting Problem[⋆]

Gabriele Di Stefano and Magnus Love Koči

Dipartimento di Ingegneria Elettrica, Università degli Studi dell'Aquila, I-67040 Monteluco di Roio - L'Aquila - Italy

Abstract

In this paper we propose a graph theoretical approach to the problem of train shunting in a railway depot. Especially during the night, the trains have to be parked in a shunting depot in such a way that the operations in the next morning can start as smoothly as possible. The general problem is very difficult and includes many subproblems. We focus on the following subproblem: how to arrange the trains in a "correct" order on the available tracks by avoiding shunting operations for outgoing trains of the morning. We discuss different cases of the problem, and we propose both algorithmic solutions and heuristic approaches.

Keywords: Shunting problem, permutation graphs, 3-uniform hypergraphs, unimodal sequences

1 Introduction

Railway operators have to solve many complex decision problems in order to achieve high quality of service at reasonable cost. One of them is the shunting problem: outside the rush hours there is surplus of rolling stock, and, especially during the night, passenger trains have to be parked in a shunting depot. A major complicating issue is the fact that trains are restricted in their movements by the railway infrastructure of the depot. A depot consists of a set of tracks and, depending on the type of tracks, the depot is called *shunting yard* or *marshalling yard*. In the first case all the tracks can be approached from one side only, in a *marshalling yard* each track can be approached from

⋆ Work partially supported by the Human Potential Program of the European Union under contract no. HPRN-CT-1999-00104 (AMORE Project)

doi:10.1016/j.entcs.2003.12.020

both sides. We assume that trains can move by themselves in both directions (e.g., this is the case of Dutch train units), and we do not deal with locomotive movements. The train arrivals and departures are regulated by a *timetable*: we will discuss both the case of arrivals and departure mixed in time and the case in which the first departure take place after the last arrival.

The literature on the shunting problem includes works of Winter and Zimmerman [14] and Blasum et al. [1] for dispatching trams. In this two papers all arrivals take place before the first departure. Gallo and Di Miele [4] discuss an extension of their work on bus dispatching in a depot taking in account mixed arrivals and departures. In [10] the shunting problem for passenger train units is faced and several aspects of this problem are discussed along with some mathematical models. Other papers [9,8] deal with the problem of rearranging cars of trains in a train depot.

Almost all the papers on the shunting problem assume or prove that the problem is hard. The main goal of this paper is to investigate the constraints that make the shunting problem hard. To this end, we do not take in consideration the size of the trains and track lengths, but we mainly focus on the constraints due to the type of depot and the arrival and departure sequences due to the timetable.

The paper is organized as follows: after a section on notations and basic concepts, in Section 3, we discuss the case in which the last train arrives before the first departing train. We show some results related to four subproblems which arise by imposing some constraints on the arrival and departure directions. In Section 4 the case of departures before the last arrival is discussed both for shunting and marshalling yards. The last section deals with some open problems.

2 Basic notations and concepts

In the following, we assume that the trains are numbered according to the departure time from the depot area. In particular, the first train leaving the depot is numbered 1, the second 2 and so on. Then the arriving sequence of n trains at the depot area is a permutation $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$ of the outgoing sequence $[1, 2, \ldots, n]$. For example, the sequence $[2, 3, 1]$ means that three trains are involved, the first incoming train will be the second to go out, the second incoming train will be the third to leave the depot, and the last one will be the first to go out. Notice that $(\pi^{-1})_i$ is the position of train i in a incoming sequence. It will be denoted π_i^{-1} and, in the above example, $\pi_2^{-1} = 1$, $\pi_3^{-1} = 2$, $\pi_1^{-1} = 3$. Moreover π^ρ denotes the *reverse sequence* of π, that is $\pi^\rho = [\pi_n, \pi_{n-1}, \ldots, \pi_1]$.

Given a sequence $S = [s_1, s_2, \ldots, s_n]$ of n distinct elements, a *subsequence of* S is a sequence $S' = [s_{i_1}, s_{i_2}, \ldots, s_{i_m}]$ such that $1 \leq i_j < i_k \leq n$ for each $j < k$.

A sequence $S = [s_1, s_2, \ldots, s_n]$ is called *unimodal* if there exists t, $1 \leq t \leq n$ such that $s_1 < s_2 < \ldots < s_t$ and $s_t > s_{t+1} > s_{t+2} > \ldots > s_n$. A unimodal sequence is called *increasing* if $t = n$, and it is called *decreasing* if $t = 1$.

Let $P_S = \{S_1, S_2, \ldots, S_k\}$ be a set of subsequences S_i, $1 \leq i \leq k$ of a sequence S of distinct elements. P_S is called a *partition* of S if each element of S belongs to exactly one subsequence in P_S. Two sequences are *disjoint* if they have no element in common.

It is useful to define some operations on sequences. We write $s \in S$ if s is an element of the sequence S, and the number of elements of S is denoted by $|S|$. If S' is a subsequence of S, then $S \setminus S'$ denotes the sequence obtained from S by removing all the elements in S'. If S and T are two sequences s.t. $|S| = n$ and $|T| = m$, then the sequence given by the *concatenation of* S *and* T, denoted by $S \cdot T$, is the sequence having S as the subsequence of the first n elements and T as the subsequence of the last m elements, and $|T \cdot S| = m+n$. If $S = [s_1, s_2, \ldots, s_n]$ is a sequence on n integers and k is an integer, then $S+k$ denotes the sequence $[s_1 + k, s_2 + k, \ldots, s_n + k]$.

Let V be a finite set of *vertices*. Let $\mathcal{P}(V)$ denote the power set of the set V, i.e., the set of all subsets of V. The pair $H = (V, \mathcal{E})$ is a *hypergraph* if $\mathcal{E} \subseteq \mathcal{P}(V)$. $\mathcal{E}$ is the set of *hyperedges* of the hypergraph. A hypergraph $H = (V, \mathcal{E})$ is called k*-uniform* if each $e \in \mathcal{E}$ has exactly k elements. A 2-uniform hypergraph is called *graph*, in this case the hyperedges are called *edges*. A k-coloring of a hypergraph $H = (V, \mathcal{E})$ is a mapping $f : V \longrightarrow \{1, \ldots, k\}$ such that no edge of H (besides singletons) has all vertices of the same color. The *chromatic number* of H, denoted by $\chi(H)$, is the minimal k, for which H admits a k-coloring.

Let $G = (V, E)$ be a graph. $V' \subseteq V$ is a *clique* in G if $(u, v) \in E$ for all $u, v \in V'$, $u \neq v$. $w(G) = \max\{|V'| : V' \subseteq V$ and V' is a clique in $G\}$ is called the *clique number* of G.

3 Problems for night depots

For the problems of this section, we generally assume that the depot topology is a marshalling one, that is each track can be approached from both sides. We do not take into account the lengths of trains and tracks. That is, we assume that each track is long enough to host the trains assigned to it. We also assume that the first departure takes place after the last arrival: This could be the case of a depot used during the night. In the next section, we discuss the case of arrivals and departures mixed in time.

The common objective of all the following problem is to use the minimum number of tracks to host all the trains without additional shunting movements during input or output operations.

By putting some restrictions on the way to approach the tracks we define four main problems:

SISO problem: Single Input, Single Output. In this case we assume that each track can be approached by one side and left from the other side, that is the tracks are used as queues. We also discuss the case in which the tracks are used as stacks (that is the inputs and the outputs of the trains are done from a single side and the other is not accessible). This is the case of a shunting yard.

DISO problem: Double Input, Single Output. In this case each track can receive trains from both sides, but the output is done from a single side. In the night, a main goal of the general shunting problem is to park trains in such a way that the operations in the next morning can start as smoothly as possible. Then the single output side in the direction of the station is a desirable requirement.

SIDO problem: Single Input, Double Output. This case is the opposite of the previous one. Even if there is no practical reason to consider this case, the study of this problem is easier w.r.t the DISO problem, because the order of the trains stored in a track will respect the order in the input sequence. It is easy to show that the SIDO and DISO problems are equivalent: we will give an evidence in Section 3.2.

DIDO problem: Double Input, Double Output. In this case, no restrictions are imposed.

3.1 Single Input Single Output (SISO)

Since the tracks are used as queues, then the assignment of trains to each track must be done in such a way that the trains in a track form an increasing sequence, that is, the first departing train must have the lowest of the numbers assigned to the trains stored in the track. This avoids the need of shuntings during the output operations.

Definition 3.1 SISO problem: Given a sequence S of n incoming trains, find a partition of S with the minimum number of increasing subsequences.

Given a sequence S, we can construct an undirected graph $G[S] = (V, E)$ in the following way:

- $V = \{s \mid s \in S\}$
- $E = \{(s_i, s_j) \mid [s_i, s_j]$ is a decreasing subsequence of $S\}$

An undirected graph G is called a *permutation graph* if there exists a sequence S such that $G \cong G[S]$. Notice that an edge (s_i, s_j) is present in $G[S]$ if s_i and s_j are in the wrong order, that is, the corresponding trains cannot be put in the same track. Moreover a decreasing subsequence of S corresponds to a clique in $G[S]$.

If the sequence S is reversed, than each pair of trains forming an increasing subsequence in S is now in decreasing order, and vice versa. Hence $G[S^\rho] = \overline{G[S]}$, which shows that the complement of a permutation graph also is a permutation graph.

Theorem 3.2 *[7] Given a sequence S of n incoming trains, the minimum number of tracks to solve the SISO problem is k if and only if $\chi(G[S]) = k$.*

Proof. Let k' be the optimum solution for the SISO problem on the sequence S, and let $k'' = \chi(G[S])$. We have to show that $k' = k''$.

The minimum number of tracks to solve the SISO problem must be larger than the longest decreasing subsequence S' in S, because no pair of trains in S' can be put in the same track. Since a decreasing sequence S' corresponds to a clique in $G[S]$, then $k' \geq \omega(G[S])$.

Now, let us suppose to have a coloring of $G[S]$. Then we can find a feasible solution to the SISO problem by putting all the trains colored with the same color in a single track. In fact, if two train s_i and s_j are colored with the same color, they must form an increasing subsequence, otherwise (s_i, s_j) is an edge of $G[S]$ and the two trains cannot be colored with the same color. As a consequence, $k' \leq \chi(G[S])$.

Then $\omega(G[S]) \leq k' \leq \chi(G[S])$. But permutation graphs are perfect graphs (see [7]), and for these graphs the chromatic number and the clique number are equal. Thus $\omega(G[S]) = \chi(G[S])$ which implies $k' = \chi(G[S]) = k''$. □

We report an algorithm (see Algorithm 1) to assign trains on tracks based on the corresponding algorithm to color a permutation graph [2,7]. Algorithm 1 uses vector $LAST$, and $LAST[i]$ holds the last train in the i-th track. The algorithm requires $O(n \log n)$ time to assign each train of a sequence S with n trains to a proper track. In fact, loop 2–7 is repeated n times and procedure *Findtrack* requires $O(\log n)$ time to find a track with a binary search.

Note that, in the case of tracks approachable only from one side (that is the SISO problem on a shunting yard), two trains $s_i, s_j \in S$, where S is the input sequence, cannot be put in the same track if $[s_i, s_j]$ is an *increasing* subsequence of S. Then in $\overline{G[S]}$ the edge (s_i, s_j) is present, as well as for all

Algorithm 1 On-line SISO for marshalling yards

```
Require: A sequence $\pi = [\pi_1, \pi_2, \ldots, \pi_n]$ of $n$ incoming trains numbered from 1 to $n$
Ensure: A track assignment for each train using the minimum number $k$ of tracks
 1: k$\leftarrow$ 0
 2: for $j = 1$ to $n$ do
 3:    i $\leftarrow$ $Findtrack$ $(\pi_j)$
 4:    Put train $\pi_j$ on track $T_i$
 5:    LAST[i] $\leftarrow$ $\pi_j$
 6:    k $\leftarrow$ $\max\{k, i\}$
 7: end for
```

Algorithm 2 Procedure **Findtrack**

```
 1: i $\leftarrow$ 1; t $\leftarrow$ k+1
 2: while i$\neq$t do
 3:    r $\leftarrow$ $\lfloor \frac{i+t}{2} \rfloor$
 4:    if $\pi_j >$ LAST[r] then
 5:       t $\leftarrow$ r
 6:    else
 7:       i $\leftarrow$ r+1
 8:    end if
 9: end while
10: return i
```

the pairs of trains which are in an increasing order in S, and there is no edge between two trains which are in decreasing order. Since $\overline{G[S]} = G[S^\rho]$, then, by Theorem 3.2, the chromatic number $\chi(G[S^\rho])$ gives the minimum number of tracks of a shunting yard to store the trains of the incoming sequence S. As a consequence, the number of tracks can be found in $O(n \log n)$ time.

By looking at Algorithm 1, we can notice that the choice of a track for the i-th train does not depend on trains arriving later. This means that the same procedure *Findtrack* can be used for the on-line version of the SISO problem, in which we have to determine the track of an incoming train, given its departure time, without knowing the entire input sequence.

3.2 DISO and SIDO problem

The DISO problem on a marshalling yard implies that the incoming trains can approach each track from both sides, but in the morning they leave the tracks from one side only. In the SIDO problem the usage of tracks is opposite, that is the incoming trains can approach the tracks from a single side, but the output can be done from both sides. In the following we will study the SIDO problem since the sequence of trains stored in a track must be a subsequence of the incoming sequence of trains. This is not true for the DISO problem. On the other hand, it is easy to show that the two problems are equivalent (see Theorem 3.13).

In the SIDO situation the sequence $S = [s_1, s_2, \dots, s_m]$ of the trains stored in a track must form a unimodal sequence. In fact, let R be the side of the track used for incoming trains and let L be the opposite side. The sequence S is such that $S = S_L \cdot S_R$, where $S_L = [s_1, s_2, \dots, s_t]$ is the subsequence of trins going out from the L side, and $S_R = [s_{t+1}, s_{t+2}, \dots, s_m]$ is the subsequence of trins going out from the R side. The trains in S_L must form an increasing sequence $s_1 < s_2 < \dots < s_t$, whereas the trains in S_R must form a decreasing sequence $s_{t+1} > s_{t+2} > \dots > s_m$, being s_m the first train leaving the track from this side. The concatenation of these two sequences is a unimodal sequence. Then the SIDO problem can be formally stated as follows:

Definition 3.3 SIDO problem: Given a sequence S of n incoming trains, find a partition of S with the minimum number of unimodal subsequences.

It is interesting to note that, every sequence of two trains can be stored in a single track: this is not true in the SISO situation. On the other hand, sequences of just three trains can require two tracks. Let $[t_1, t_2, t_3]$ be a sequence of three incoming trains, and let us suppose they are stored in a single track. If $t_1 > t_2$ and $t_2 < t_3$ the train t_2 must leave the depot before the other two: impossible without shunting operations. As consequence, we can store in a single track at most two of the three trains. To keep this kind of constraints, we introduce the notion of *valley hypergraph.*

Definition 3.4 Given a sequence $S = [s_1, s_2, \dots, s_n]$ of n distinct integers, the *valley hypergraph induced by* S is the hypergraph $H_S = (V, \mathcal{E})$ where $V = \{s \mid s \in S\}$ and $\{s_i, s_j, s_k\} \in \mathcal{E}$ if and only if $s_i > s_j$, $s_j < s_k$, and $i < j < k$.

Lemma 3.5 *Given a sequence S of n incoming trains, the SIDO problem is solvable using k tracks if and only if the valley hypergraph $H_S = (V, \mathcal{E})$ is colorable using k colors.*

Proof. $\Longrightarrow$*:* If the SIDO problem is solvable using k tracks, then there exist k unimodal subsequences $S_1, S_2, \dots, S_k$ that form a partition of S. We color all the trains in the same track with the same color, that is, if $t \in S_i$ we color t with color i, for each $i = 1, \dots, k$.

By contradiction, let us suppose that $\{s_h, s_i, s_j\} \in E$, $h < i < j$ and that

s_h, s_i, and s_j are colored with the same color. This implies that the trains s_h, s_i, and s_j are stored in the same track. Then the sequence $[s_h, s_i, s_j]$ is a subsequence of a unimodal sequence and hence it is unimodal. This implies that s_h, s_i, and s_j cannot form a valley, contradicting the hypothesis $\{s_h, s_i, s_j\} \in E$.

$\Longleftarrow$*:* Since H_S is colorable with k colors, then there exist k sets $T_1, T_2, \ldots, T_k$ of trains colored with the same color. Let T_c be the set colored with the color c and S_c the corresponding subsequence of S. Then S_c does not contain valley configurations of trains. This implies that S_c is unimodal. Since we can repeat this proof for each color, then there exist $S_1, S_2, \ldots, S_k$ unimodal subsequences that form a partition of S.

□

The complexity of the SIDO problem is open, but we will show a greedy algorithm to color a valley hypergraph. We first show a property of unimodal subsequences by using the *Pigeonhole Principle*:

Definition 3.6 *Pigeonhole Principle*: Let A be a set. Let B be a finite set. If $f : A \longrightarrow B$ and $|A| > |B| \geq 1$, then there exists $a_1 \neq a_2 \in A$ such that $f(a_1) = f(a_2)$.

Lemma 3.7 *In any sequence of* $\frac{k(k-1)}{2} + 1$ *distinct integers there exists a unimodal sequence of* k *integers.*

Proof. Let $S = [s_1, s_2, \ldots, s_{\frac{k(k-1)}{2}+1}]$ be a sequence of distinct integers. By contradiction, every subsequence of S with k or more elements is not unimodal. We assign a pair of integers (x_i, y_i) to each element s_i of S in such a way that $x_i \geq 1$ is the length of the longest increasing subsequence of S ending in s_i, whereas $y_i \geq 1$ is the length of the longest decreasing subsequence of S starting with s_i. Then $x_i + y_i - 1$ is the length of the longest unimodal subsequence of S having s_i as largest element. By contradiction hypothesis, the following inequality holds.

$$x_i + y_i - 1 < k \qquad \text{for each } i$$

Since x_i and y_i are positive integers, this implies that there are at most $\frac{k(k-1)}{2}$ distinct pairs (x_i, y_i).

Let $A = \{s_j | s_j \in S\}$ and let $B = \{(x_j, y_j) \mid (x_j, y_j)\text{is assigned to } s_j \in S\}$. Let $f : A \longrightarrow B$ be the function that represents the above assignment. Since $|A| > |B|$, then, by using the Pigeonhole Principle, there exist s_i and s_j such that $f(s_i) = f(s_j)$ for some $i \neq j$, i.e., $(x_i, y_i) = (x_j, y_j)$. Without loss of generality, let us suppose $i < j$. Then, either $s_i < s_j$ or $s_i > s_j$. If $s_i < s_j$, then $x_i < x_j$ by definition of (x_i, y_i) and (x_j, y_j), a contradiction. If $s_i > s_j$, then $y_i > y_j$, a contradiction. □

Theorem 3.8 *Let H_S be a valley hypergraph such that $|S| = n$. If $\chi(H_S) \geq k$ then $n \geq \frac{k(k+1)}{2}$.*

Proof. We prove this theorem by induction on the number k of colors. Case $k = 1$: If $\chi(H_S) \geq 1$, we have to show that $n \geq 1$, but this is obvious. Case $k > 1$: By induction, we assume that if $\chi(H_S) \geq k-1$ then $n \geq \frac{k(k-1)}{2}$. We show the inductive step by contradiction, that is, we assume $\chi(H_S) \geq k$ and $n < \frac{k(k+1)}{2}$.

From the inductive and contradiction hypotheses, we obtain $\frac{k(k-1)}{2} \leq n < \frac{k(k+1)}{2}$. Then $n = \frac{k(k-1)}{2} + p$, where p is an integer such that $0 \leq p < k$.

Case $1 \leq p < k$: In this case, by Lemma 3.7, there must exist a unimodal subsequence S' of S with k elements. Let $T = S \setminus S'$, then the valley hypergraph H_T has $|T| = n - k < \frac{k(k+1)}{2} - k = \frac{k(k-1)}{2}$ vertices. Then, by the induction hypothesis, $\chi(H_T) < k-1$. Hence H_T is colorable with at most $k-2$ colors and then, since the elements of S' are colorable with a single color, H_S is colorable with $k-1$ colors, a contradiction.

Case $p = 0$: In this case, by Lemma 3.7, there must exist a unimodal subsequence S' of S with $k-1$ elements. Let $T = S \setminus S'$, then the valley hypergraph H_T has $|T| = n - (k-1) = \frac{k(k-1)}{2} - (k-1) < \frac{k(k-1)}{2}$ vertices. Then, by the induction hypothesis, H_T is colorable with at most $k-2$ and then H_S is colorable with $k-1$ colors, a contradiction.

□

Corollary 3.9 *The SIDO problem on a sequence of n trains is solvable with at most $\left\lfloor \frac{\sqrt{8n+1}-1}{2} \right\rfloor$ tracks.*

Proof. Let S be the sequence of incoming trains and let $k = \left\lfloor \frac{\sqrt{8n+1}-1}{2} \right\rfloor$. By contradiction, we assume that k tracks are not sufficient. By Lemma 3.5, this means that the valley hypergraph H_S is not colorable with k colors, that is $\chi(H_S) \geq k+1$. By Theorem 3.8, we obtain $n \geq \frac{(k+1)(k+2)}{2}$. But, by using simple algebra, we obtain

$$n \geq \frac{(k+1)(k+2)}{2} > \frac{\frac{\sqrt{8n+1}-1}{2}\left(\frac{\sqrt{8n+1}-1}{2}+1\right)}{2} = n$$

a contradiction. □

Corollary 3.9 gives an upper bound to the number of tracks needed to solve the SIDO problem on n trains. The following theorem shows that this bound is tight.

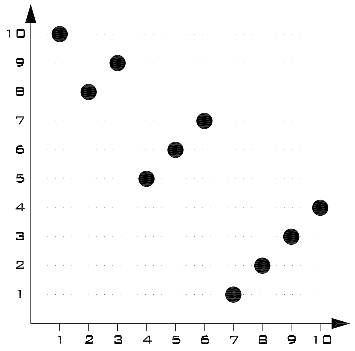


Fig. 1. The sequence $S_4 = [10, 8, 9, 5, 6, 7, 1, 2, 3, 4]$ of trains requiring four tracks in the SIDO problem

Theorem 3.10 *Let m be an arbitrary large intege. There always exists a valley hypergraph H_S, such that $|S| > m$ and $\chi(H_S) = \left\lfloor \frac{\sqrt{8|S|+1}-1}{2} \right\rfloor$.*

Proof. We show this theorem when $|S| = n = \frac{k(k+1)}{2}$, where k is an arbitrary positive integer such that $n > m$. In this special case $\left\lfloor \frac{\sqrt{8n+1}-1}{2} \right\rfloor = k$. Let S_i, $i \geq 1$ be a sequence defined in the following way:

$$S_i = \begin{cases} [1] & \text{if } i = 1 \\ (S_{i-1} + i) \cdot [1, 2, \ldots, i] & \text{otherwise} \end{cases}$$

Figure 1 shows the sequence S_4. We show that the valley hypergraph H_{S_k} has $|S_k| = \frac{k(k+1)}{2}$ elements and requires k colors to be colored. By induction on k, we assume that $H_{S_{k-1}}$ has $\frac{k(k-1)}{2}$ elements and requires $k-1$ colors to be colored. The basic step of the induction is obvious.

It is easy to see that k colors are sufficient to color H_{S_k}. In fact, S_k is formed by the concatenation of k increasing (and then unimodal) subsequences. It is sufficient to use a different color for each subsequence.

By contradiction, let us suppose that $k-1$ colors are enough to color H_{S_k}.

Let x and y be two vertices of a 3-uniform hypergraph $H = (V, \mathcal{E})$ and let $T_{xy} = \{z \mid \{x, y, z\} \in \mathcal{E}\}$ be the set of all vertices that form an hyperedge with x and y. Let $H(T_{xy})$ be the sub-hypergraph of H induced by the vertices in T_{xy}. If $\chi(H) = \chi(H(T_{xy}))$ then x and y must be colored with two different colors, otherwise there exists a hyperedge in H with three nodes (x, y, and a node in T_{xy}) colored with the same color.

The above property can be applied to H_{S_k}. Let x and y two vertices in the subsequence $[1, 2, \ldots, k]$ of S_k. Then all the nodes z such that $\{x, y, z\}$ is a hyperedge in H_{S_k} are exactly the nodes of the subsequence $S_{k-1} + k$. Let

T be the set of these nodes and let $H(T) \equiv H_{S_{k-1}+k}$ be the sub-hypergraph of H_{S_k} induced by the nodes in T. Notice that the valley hypergraph induced by $H(T)$ is isomorph to the valley hypergraph induced by S_{k-1}, that is $H_{S_{k-1}}$. Then, by induction hypothesis, $\chi(H(T)) = \chi(H_{S_{k-1}}) = k-1$, and x and y must be colored with different colors since $\chi(H_{S_k}) = k-1$ by contradiction hypothesis.

Hence, every pair of nodes in the subsequence $[1, 2, \ldots, k]$ of S_k must be colored with two different colors. But this means that k colors are needed, a contradiction with the hypothesis that $k-1$ colors are enough to color H_{S_k}.□

Algorithm 3 Track assignment for the SIDO problem

Require: A sequence S of n incoming trains

Ensure: A track assignment for each train using at most $\left\lfloor \frac{\sqrt{8n+1}-1}{2} \right\rfloor$ tracks

```
1: Set $t = 1$
2: **while** $S \neq [\,]$ **do**
3:     $S' = \textbf{LongestUnimodal}(S)$
4:     Assign each element in $S'$ to the $t$-th track.
5:     Set $S = S \setminus S'$
6:     Set $t = t + 1$
7: **end while**
```

On the basis of the previous results, it is possible to show that there exists a greedy algorithm that uses at most $\left\lfloor \frac{\sqrt{8n+1}-1}{2} \right\rfloor$ tracks. Given a sequence S of trains, the Algorithm 3 finds the longest unimodal sequence S' in S and assigns each train in S' to the first available track. Then, iteratively, it repeats the operation on the sequence of the remaining trains. Procedure *LongestUnimodal*, shown in Algorithm 4, follows the same technique used in Lemma 3.7: it associates a pair of integers (x_i, y_i) to each element s_i of S in such a way that $x_i \geq 1$ is the length of the longest increasing subsequence of S ending in s_i, whereas $y_i \geq 1$ is the length of the longest decreasing subsequence of S starting with s_i (see lines 1–9). Then it finds the biggest element of the longest subsequence S' (lines 10–11), and all the other elements of S' (those on the left, lines 12–17, and those on the right, lines 18–23).

Theorem 3.11 *Algorithm 3 solves the SIDO problem on a sequence of n trains using at most $\left\lfloor \frac{\sqrt{8n+1}-1}{2} \right\rfloor$ tracks.*

Proof.

The number of tracks used by Algorithm 3 is equal to the number of times the cycle (lines 2–7) is repeated. In what follows we calculate this number.

Algorithm 4 Procedure **LongestUnimodal**

```
Require: A sequence $S = [s_1, s_2, \ldots, s_n]$, $n \geq 1$
Ensure: A longest unimodal subsequence $S'$ of $S$
    Set $(x_i, y_i)$ to $(1,1)$, for each $i = 1, 2, \cdots, n$
    for $i = 2, 3, \cdots, n$ do
      Set $X^i = \{x_j | s_j \in [s_1, s_2, \ldots, s_{i-1}]\text{and } s_j < s_i\}$
      Set $x_i = \max(X^i) + 1$          /* $\max(X^i) = 0$ if $X^i$ is empty */
 5: end for
    for $i = n-1, n-2, \cdots, 1$ do
      Set $Y^i = \{y_j | s_j \in [s_{i+1}, s_{i+2} \ldots, s_n]\text{and } s_j < s_i\}$
      Set $y_i = \max(Y^i) + 1$          /* $\max(Y^i) = 0$ if $Y^i$ is empty */
    end for
10: Let $s_t \in S$ such that $x_t + y_t \geq x_i + y_i$, for each $i = 1, 2, \cdots, n$
    Set $S' = [s_t]$; $x = x_t$; $y = y_t$
    for $i = t-1, t-2, \cdots, 1$ do
      if $x_i = x - 1$ then
        Set $S' = [s_i] \cdot S'$
15:     Set $x = x - 1$
      end if
    end for
    for $i = t+1, t+2, \cdots, n$ do
      if $y_i = y - 1$ then
20:     Set $S' = S' \cdot [s_i]$
        Set $y = y - 1$
      end if
    end for
```

Let $t_i = \frac{i(i+1)}{2}$. We want to show that if $n < t_{i+1}$ then the loop is repeated at most i times. When $n < t_1$, that is the sequence is empty, the loop is not executed, and, correctly, no track assignment is performed.

By induction, we assume that, if $n < t_i$ then the cycle is repeated at most $i - 1$ times.

If $t_i < n < t_{i+1}$, by Lemma 3.7, S contains a unimodal subsequence of length $i + 1$. Since Algorithm 3 finds the longest unimodal subsequence S' at Step 3, then $|S'| \geq i+1$. This means that $|S \setminus S'| \leq n-(i+1) < t_{i+1}-(i+1) = t_i$. Then, by inductive hypothesis, the loop is repeated at most $i - 1$ times for the sequence $S \setminus S'$, and hence at most i times for S.

If $n = t_i$, by Lemma 3.7, S contains a unimodal subsequence of length i, and Algorithm 3 finds the longest unimodal subsequence S' at step 3, then $|S'| \geq i$. This means that $|S \setminus S'| \leq n - i \leq t_i - i = t_{i-1}$. Then, by inductive hypothesis, the cycle is repeated at most $i-1$ times for the sequence $S \setminus S'$, and hence i times for S.

Now, if $t_k \le n < t_{k+1}$, then on one hand the loop is repeated at most k times and on the other hand $\frac{k(k+1)}{2} \le n$, thus $k \le \frac{\sqrt{8n+1}-1}{2}$. $\square$

The complexity of Algorithm 3 is $O(n^{2.5})$ because the *Findtrack* procedure takes $O(n^2)$ time, and it is repeated $O(\sqrt{n})$ times, as stated in the above result.

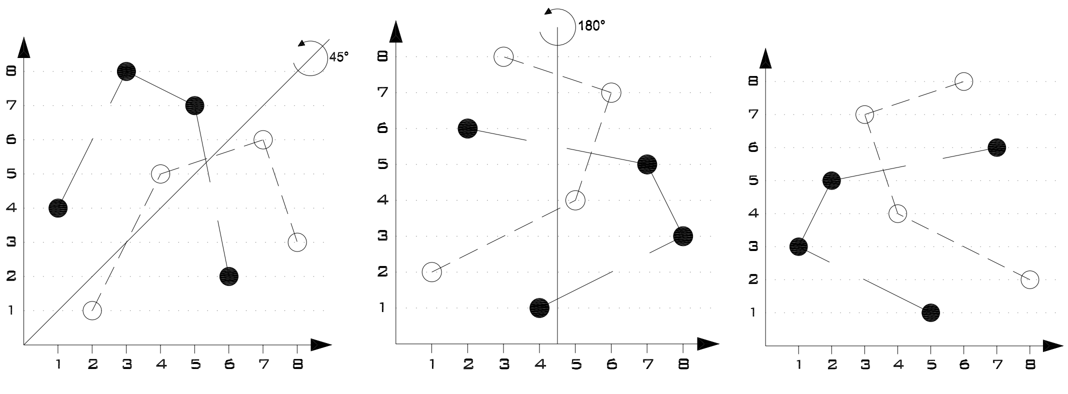
In what follows, we show the equivalence of the SIDO and DISO problems. In the DISO problem, since the output is from a single side of the track, the trains are stored in each track in such a way that they form an increasing sequence starting from the first train going out. This means that, as soon as we put the first train t in an empty track, all the other trains numbered with a smaller number w.r.t. t must be put between t and the output in a decreasing order starting from t, and all the other trains numbered with a larger number must be put behind t in an increasing order (starting from t). As consequence, in a DISO problem a sequence S of trains can be put in a single track if it is an *arrow sequence*, i.e. if $S = [t] \cdot T$ and T is partitionable into two sequences S' and S'' such that S' is an increasing sequence of elements larger than t, and S'' is a decreasing sequence of elements smaller than t. Figure 2c shows two arrow subsequence of sequence $[3,5,7,4,1,8,6,2]$, namely $[3,5,1,6]$ and $[7,4,8,2]$. The DISO problem can be formally stated as follows.

Definition 3.12 DISO problem: Given a sequence S of n incoming trains, find a partition of S with the minimum number of arrow subsequences.

The following theorem states the equivalence between the SIDO and the DISO problem.

Theorem 3.13 *Let π be a permutation of n trains, numbered $1,2,\ldots,n$. The SIDO problem on the sequence π can be solved by using k tracks if and only if the DISO problem on the sequence $(\pi^{-1})^\rho$ can be solved by using k tracks.*

Proof. $\Longrightarrow$*:* The SIDO problem requires to find a partition of the sequence π into unimodal subsequences. We prove the theorem by showing that a unimodal subsequence S in π corresponds to an arrow subsequence in $(\pi^{-1})^\rho$ with the same number of elements. If we represent π like a function graph (see Fig 2a), then π^{-1} can be obtained by mirroring the function graph along the line positioned at 45 degrees (see Fig 2b). It is easy to see that a unimodal subsequence in π corresponds to a mirrored arrow subsequence in π^{-1}, and then to an arrow subsequence in $(\pi^{-1})^\rho$ (see Fig 2c). Hence if we know a solution for the SIDO problem on π, we can easily find a solution for the DISO problem on $(\pi^{-1})^\rho$.



(a) π (b) π^{-1} (c) $(\pi^{-1})^{\rho}$

Fig. 2. Transformation from a $\pi = [4,1,8,5,7,2,5,3]$ sequence to a $(\pi^{-1})^{\rho} = [3,5,7,4,1,8,6,2]$ sequence.

$\Longleftarrow$*:* On the other hand, if we know a solution for the DISO problem on $\tau = (\pi^{-1})^{\rho}$, that is a partition of τ into k arrow subsequences, we can easily find a solution for the SIDO problem on π, that is a partition of π into k unimodal subsequences, since $\pi = (\tau^{\rho})^{-1}$.

□

3.3 Double Input Double Output (DIDO)

In this section we just model the DIDO problem as a coloring problem of 4-uniform hypergraphs.

The DIDO problem on a marshalling yard implies that the incoming trains can reach and leave the tracks from both sides.

In the DIDO situation the trains stored in a track must form a unimodal sequence as well as in the SIDO problem. The first train t stored in the track represents a point that divides the unimodal sequence into two subsequences of trains. In the input sequence, one of these two subsequences is unimodal, the other is decreasing. More formally, in a DIDO problem, a not empty sequence S of trains can be put in a single track if $S = [t] \cdot T$ and T is partitionable into two sequences S' and S'' such that S' is a decreasing sequence of elements smaller than t, and $[t] \cdot S''$ is a unimodal sequence. We call S a *ud-sequence* (*u*nimodal and *d*ecreasing sequence). Then the DIDO problem can be formally stated as follows.

Definition 3.14 DIDO problem: Given a sequence S of n incoming trains, find a partition of S with the minimum number of ud-subsequences.

Notice that every sequence of three trains can be stored in a single track: this is not true in the SISO, SIDO, and DISO problems. On the other hand, sequences of four trains can require two tracks. For example, let $S = [3, 1, 2, 4]$: it is not a ud-sequence. We store train 3 in an empty track, and, in the same track, we can store trains 1 and 2, necessarily both adjacent to train 3. If we put the train 4 in the same track, then either train 1 or train 2 cannot leave the track before train 4. The sequences $[1, 3, 2, 4], [1, 4, 2, 3], [3, 1, 2, 4]$, and $[4, 1, 2, 3]$ are the only sequences with four elements that are not ud-sequences. We call them *nud-sequences* (*n*ot *ud*-sequences).

Definition 3.15 Given a sequence $S = [s_1, s_2, \ldots, s_n]$ of n distinct integers, the *nud-hypergraph induced by* S is the hypergraph $H_S = (V, \mathcal{E})$ where $V = \{s \mid s \in S\}$ and $\{s_h, s_i, s_j, s_k\} \in \mathcal{E}$ if and only if $h < i < j < k$ and $[s_h, s_i, s_j, s_k]$ is a nud-subsequence of S.

Lemma 3.16 *Given a sequence S of n incoming trains, the DIDO problem is solvable using k tracks if and only if the nud-hypergraph $H_S = (V, \mathcal{E})$ is colorable using k colors.*

Proof. $\Longrightarrow$*:* If the DIDO problem is solvable using k tracks, then there exist k ud-subsequences $S_1, S_2, \ldots, S_k$ that form a partition of S. We color all the trains stored in the same track with the same color, that is, if $t \in S_i$ we color t with color i, for each $i = 1, \ldots, k$.

By contradiction, let us suppose that $\{s_h, s_i, s_j, s_k\} \in \mathcal{E}$, $h < i < j < k$ and that s_h, s_i, s_j, and s_k are colored with the same color. Then the trains s_h, s_i, s_j, and s_k are stored in the same track, forming a unimodal sequence in that track. This implies that $[s_h, s_i, s_j, s_k]$ is a ud-subsequence of S, contradicting the hypothesis $\{s_h, s_i, s_j, s_k\} \in \mathcal{E}$.

$\Longleftarrow$*:* Since H_S is colorable with k colors, then there exist k sets $T_1, T_2, \ldots, T_k$ of trains colored with the same color. Let T_c be the set colored with the color c and let S_c be the corresponding subsequence of S. Then S_c does not contain nud-subsequences of trains. This implies that S_c is a ud-sequence and the trains can be stored in a single track. Since we can repeat this proof for each color, then there exist $S_1, S_2, \ldots, S_k$ ud-subsequences that form a partition of S.

□

4 Problems for day depots

In this section we face the SISO problem by removing the constraint that the first outgoing train leaves the depot after the last incoming train. We call this problem *General SISO*, denoted GSISO, and we will discuss this problem

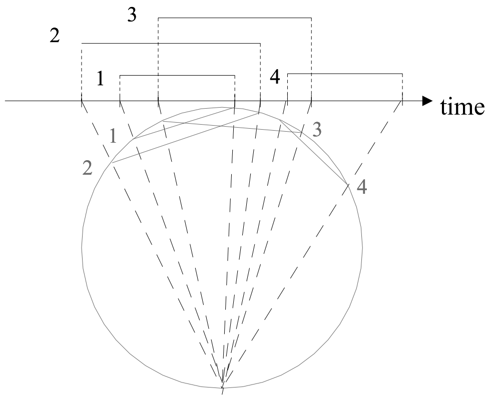


Fig. 3. A set of intervals and the corresponding chords in a circle.

both for marshalling yards and for shunting yards.

We start assuming a shunting yard. We can model this situation with a set of intervals on an oriented line. Each interval is associated to a train and represents the interval of time that the train have to spend in the depot. We number the trains (and then the intervals) as usual: the first train leaving the depot is numbered 1, the second 2 and so on. Figure 3 represents the model, and it also shows that there is a one-to-one correspondence between each interval and a chord in a cycle. Note that in this model, the constraint that the first outgoing train leaves the depot after the last incoming train, is equivalent to impose that all the intervals share at least a point.

We say that two intervals *overlap* if they intersect but neither properly contains the other. It is easy to see that if two intervals overlap, the corresponding trains cannot be put in the same track of a shunting yard. By denoting $\mathcal{I}$ the set of intervals and $I_j \in \mathcal{I}$ the interval associated to train j, for each j, we can construct an undirected graph $G[\mathcal{I}] = (V, E)$ in the following way:

- $V = \{j \mid I_j \in \mathcal{I}\}$
- $E = \{(i, j) \mid I_i \text{ and } I_j \text{ overlap}\}$

$G[\mathcal{I}]$ is called *overlap graph.* The following lemma shows a correspondence between the GSISO and the coloring of an overlap graph.

Lemma 4.1 *Let $\mathcal{I}$ be a family of intervals associated to a set of trains T. The GSISO problem on a shunting yard for the set T can be solved with k tracks if and only if there exists a k-coloring of $G[\mathcal{I}]$.*

Proof. $\Longrightarrow$*:* We associate to each track a different color and then we color each train stored in a track with the color associated to that track. Since connected vertices i and j of $G[\mathcal{I}]$ are assigned to different tracks, they receive different colors.

$\Longleftarrow$: Conversely, given a proper coloring of $G[\mathcal{I}]$ using colors $1, 2, \ldots, k$. If the color of the vertex x is i, then the train x is assigned to the i-th track. Suppose this strategy is unsuccessful. There must be a bottleneck in some track: this track has a pair of trains x and y stored in increasing order. Without loss of generality, let $x < y$, then x entered before y. But, since $x < y$, train x must leave the depot before train y, and this means that I_x overlap I_y. As a consequence x and y are adjacent in $G[\mathcal{I}]$, and they are both colored the same, a contradiction. $\Box$

An immediate consequence of Lemma 4.1 is that an optimum solution for a GSISO problem on a shunting yard is equivalent to finding the chromatic number of an overlap graph. But overlap graphs are equivalent to *circle graphs*, i.e., intersection graphs of chords in circles (the proof can be obtained with the projection method suggested by Gavril [5], see Figure 3), and in [6] it is reported that the chromatic number problem for circle graphs is NP-complete. The 3-coloring of a circle graph (equivalent to solving the GSISO problem on a shunting yard with 3 tracks) is in P, whereas the 4-coloring is NP-complete [13].

As regards the GSISO problem for marshalling yards, note that it makes no difference whether we assume arrivals and departures either mixed or not mixed in time. As a consequence the GSISO problem for marshalling yards can be solved in $O(n \log n)$ time by using Algorithm 1 on the sequence of incoming trains.

5 Open problems

The complexity of the DISO, SIDO and DIDO problems still remains open: they are equivalent to coloring particular 3-uniform and 4-uniform hypergraphs. The coloring of a general 3-uniform graph is hard and Lovász [11] proved that is NP-hard to determine whether a 3-uniform hypergraph is 2-colorable. It is also hard to approximate the chromatic number of a (hyper)graph within a factor of $n^{1-\epsilon}$ [12].

Other models should be studied when some other factors are introduced to the shunting problem. For example, the influence of the length of both tracks and trains is absolutely not marginal. Another factor is the type of the trains: at departure time it is required a train of a certain type and not a specific train, as assumed above. Then a matching among the arriving trains and the departing trains is needed.

Acknowledgments

We wish to thank Sabine Cornelsen for the useful discussions and suggestions given during her permanence at the Electrical Engineering Department of the University of L'Aquila, Italy.

References

[1] U. Blasum, M.R. Bussieck, W. Hochstättler, C. Moll, H. Scheel, and T. Winter. Scheduling trams in the morning. *Mathematical Methods of Operations Research*, 49(1): 137–148, 2000.

[2] A. Brandstädt, On improved time bounds for permutation graph problems. *Lecture Notes in Computer Science*, 657, 1–10,1993.

[3] A. Brandstädt, V.B. Le, J.P. Spinrad *Graph classes: a survey*, SIAM, 1999.

[4] G. Gallo and F. Di Miele. Dispatching buses in parking depots, *Transportation Science*, 35(3): 322–330, 2001.

[5] F. Gavril. Algorithms for a maximum clique and a maximum independent set of a circle graph, *Networks*, 3: 261–273, 1973.

[6] M.R. Garey, D.S. Johnson *Computers and intractability*, Freeman, 1979.

[7] M.C. Golumbic *Algorithmic graph theory and perfect graphs*, Academic press, 1980.

[8] S. He, R. Song, and S.S. Chaudhry. Fuzzy dispatching model and genetic algorithms for railyards operations. *European Journal of Operations Research*, 124: 307–331, 2000.

[9] E. Dahlhaus, P. Horak, M. Miller, J.F. Ryan, The train marshalling problem, *Discrete Applied Mathematics*, 103 (1–3): 41–54, 2000

[10] R. Freling, R.M. Lentink, L.G. Kroon, D. Huisman, Shunting of Passenger Train Units in a Railway Station. Ecomometric Institute Report 277, Erasmus University Rotterdam, Econometric Institute, 2002

[11] L. Lovász, Coverings and colorings of hypergraphs, *Proc. 4-th S.E. Conf. on Combinatorics, Graph Theory and Computing*, 3–12, 1973.

[12] M. Krivelevich, B. Sudakov, Approximate coloring of uniform hypergraphs, *Proc. ESA: Annual European Symposium on Algorithms*, LNCS 1461, 477–489,1998

[13] W. Unger, On the k-coloring of Circle Graphs, *Lecture Notes in Computer Science*, 294: 61–72, 1988.

[14] T. Winter and U.T. Zimmermann. Real-time dispatch of trams in storage yards. *Annals of Operations Research*, 96: 287–315,2000.