Train Scheduling on a Unidirectional Path

Apoorv Garg[1] and Abhiram G. Ranade[2]

1 **Department of Computer Science and Engineering, Indian Institute of Technology Bombay, Mumbai, India**
apoorv.garg@gmail.com

2 **Department of Computer Science and Engineering, Indian Institute of Technology Bombay, Mumbai, India**
ranade@cse.iitb.ac.in

Abstract

We formulate what might be the simplest train scheduling problem considered in the literature and show it to be NP-hard. We also give a log-factor randomised algorithm for it. In our problem we have a unidirectional train track with equidistant stations, each station initially having at most one train. In addition, there can be at most one train poised to enter each station. The trains must move to their destinations subject to the constraint that at every time instant there can be at most one train at each station and on the track between stations. The goal is to minimise the maximum delay of any train. Our problem can also be interpreted as a packet routing problem, and our work strengthens the hardness results from that literature.

1998 ACM Subject Classification F.2.2 Sequencing and scheduling, G.2.1 Combinatorial algorithms, G.2.2 Network problems

Keywords and phrases Combinatorial optimisation, Train scheduling, Max-delay minimisation, Complexity analysis, Approximation algorithm

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2017.29

1 Introduction

In this paper, we formulate a very simple train scheduling problem, show its NP-hardness, and give a randomised log-factor approximation algorithm.

Train scheduling is an extensively researched area (see, for example, the recent surveys of [1, 7, 19, 21]). The major model used is as follows. We are given a graph in which vertices represent stations and edges represent tracks. Initially, each station may hold one or more trains, which are to be moved to specified stations using specified paths. Under the standard signalling regime, on each edge there can be at most one train, and it takes some specified finite time for a train to cross an edge. Additionally, there exists a buffering limitation – each station is capable of holding no more than a specified number of trains. The goal is to move the trains such as to minimise the *makespan* (maximum completion time), *flow-time* (total completion time), or *max-delay* (maximum delay suffered by any train). This abstract problem is also studied in the packet routing literature with trains, stations, and tracks replaced by packets, network nodes, and communication links.

The problem as defined above is known to be NP-complete in various versions. See [20] for minimising the makespan, assuming unbounded buffering at each node, even when the graph is a tree. See [5] for minimising the makespan or max-delay even to a constant factor, assuming bounded buffering at nodes, for levelled directed networks in which packets move from the lowest numbered level to the largest numbered level.

37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2017).
Editors: Satya Lokam and R. Ramanujam; Article No. 29; pp. 29:1–29:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

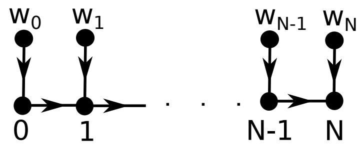

Figure 1 *The model network*

For general networks, constant factor approximation algorithms are known for minimising the makespan even with constant number of buffers at each node [14, 15, 24, 22]. For directed as well as undirected trees a 2-approximation of the optimal makespan can be obtained if we have an unbounded number of buffers in each node [20]. For unidirectional rings, in-trees, and out-trees, the optimal makespan can be obtained with unbounded buffers [16]. One recent study analyses the computational complexity of a specific train scheduling problem [10]. It models the problem of scheduling, with minimum makespan, several trains from opposite sides along a single bi-directional track with unbounded capacity at intermediate stations where trains can pass and cross each other. The problem is shown to be pseudo-polynomially solvable with equal train-speeds; however, with different speeds, it can be translated to a job-shop scheduling problem that is already strongly NP-hard.

Our concern in this paper is scheduling trains on a directed path, with trains allowed to enter or exit the path at any station. We consider minimising the max-delay, which we believe is more appropriate for train scheduling. The motivation for our study is twofold. First, large train networks are often broken into smaller networks for the purpose of managing them. These smaller networks often consist of a major trunk route with trains entering and exiting the route from and to branch lines. Each direction of the route is like the path network we consider. In addition, we are interested in a path network also because it is presumably the simplest network possible. Indeed, we further simplify the network – we assume that the inter-station distances, as well as the train speeds, are identical. We feel that we should figure out good theory for this elementary model before moving on to more complex ones. Finally, we note that there is also a large amount of experimental work on train scheduling using simulation, heuristics, integer linear programming, game theory, etc. [2, 3, 4, 6, 8, 11, 12, 17, 18, 23]. Our interest, as explained earlier, is different.

Outline of the paper is as follows. In Section 2, we formally define our problem. We also define in Section 3 a *chain-hole view* of the movement of trains, which is useful in the exposition of our lower and upper bounds. In Section 4, we show that finding a schedule with minimum max-delay is NP-hard. In Section 5, we present a randomised algorithm that schedules trains such that the max-delay is within a log-factor of the optimal. Section 6 concludes with directions for future work.

2 The Train Scheduling problem

We consider the network shown in Figure 1. It consists of :

1. The *line* – a sequence of $N+1$ stations labelled $0, 1, \ldots, N$, and unit-length directed links $(s, s+1)$ connecting every pair of consecutive stations s and $s+1$.
2. The *branches* – an *outer* (where a train waits before entering the line) w_s corresponding to each station s, and a unit-length link (w_s, s) connecting w_s to s.

Every station and outer has a capacity to hold at most one train at a time. A train takes unit time to move from one station to the next, or from an outer to its corresponding station. When there is no train at a station s, we say that there is a *hole* at s. When a train reaches

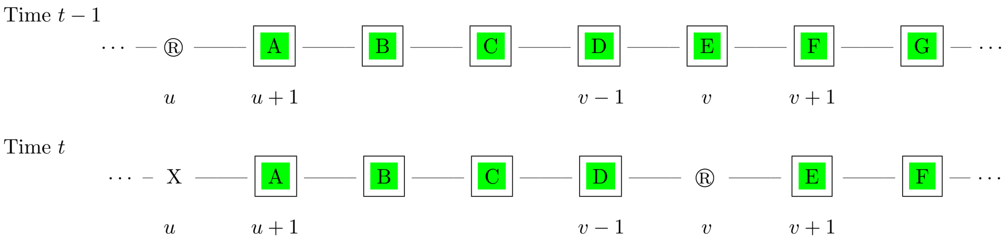

Figure 2 *Example of a hole-jump*
Boxes represent trains, circles represent holes. At time $t-1$, there is a hole R at station u. During step t, train E moves from station v to $v+1$, while the trains A, B, C, and D wait at their respective stations $u+1$ to $v-1$. The hole thus appearing at station v at time t is considered to be the same as was present at station u at time $t-1$. We say that the hole jumped from u to v in step t.

its destination, it immediately *vanishes* (exits), leaving a hole at that station. Every train that is initially at a station is called an *internal train*, while every train that is initially at an outer is called an *external train*.

Path of a train consists of all nodes and links it visits during its journey, including the origin and the destination. Note that an external train has to *enter* the line (i.e., move from the outer to its corresponding station) before it can move on the line towards its destination. Therefore, while path-length of an internal train is just the distance from its origin to its destination, that of an external train is one unit more than the distance from its entry station to its destination. The event of an external train entering the line will be referred to as an *entry*, to distinguish it from a *movement* which will refer to the event of a train moving on the line from one station to the next.

An instance is defined by specifying (i) the number N, and (ii) destination of the train, if any, placed initially at each station and outer; the destination must of course be downstream of the initial position. In any schedule for movement of the trains to their destinations, the amount of time a train remains stationary is said to be its *delay*. The required output is a schedule such that *max-delay* – the maximum among the delays – is minimised.

The t^{th} *step* of the schedule denotes the unit time duration $(t-1, t]$. The *entry time* of an external train is the time when it entered the line. *Last-entry-time* of the schedule is the last time instant when some external train entered the line, i.e., the maximum among the entry times of external trains. Without loss of generality, we assume that after the last-entry-time, all the trains proceed to their destinations without further delays.

▶ **Theorem 1.** *In any schedule,* $(\textit{last-entry-time} - 1) \leq \textit{max-delay} \leq \textit{last-entry-time}$

Proof. No train waits after the last-entry-time, say T. Hence, the max-delay can be at most T. If an external train has suffered the maximum delay, then the max-delay is $T-1$. If every train that suffered the maximum delay is an internal train, then the max-delay is T. ◀

Hence, from now on, we will worry about minimising the last-entry-time.

3 Chain-Hole view of schedules

An external train can only enter a hole. Thus, it is useful to understand holes :

Pre-existing holes. Initially, as part of the input, we are given some *pre-existing* holes on the line. In addition, it will be convenient to assume that we also have stations $-1, -2, \ldots$ upstream of station 0, each having a pre-existing hole; we will call them as *external holes*. Clearly, these *imaginary* stations and holes cannot affect the movement of trains.

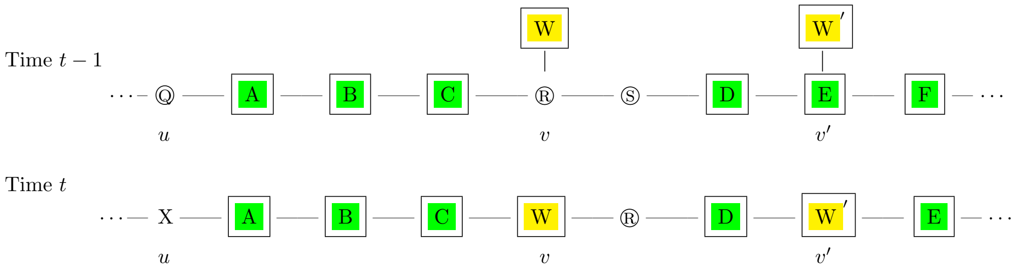

Figure 3 *External trains enter by filling holes*
In step t, hole Q jumps across four links, from station u to station $v = u + 4$, and is immediately filled by an external train W that enters the line at station v simultaneously with the jump of Q. In the same step, another external train W′ enters the line at a station v', filling a hole S which has simultaneously jumped to v' from station $v' - 2 = v + 1$. Hence, the trains A, B, C, and D have to wait during this step, while the trains E and F as well as the hole R must move.

Hole creation. A hole is created when a train moves into its destination station and vanishes.
Hole destruction. A hole is destroyed when it is *filled* by an external train.
Hole movement (jump). Suppose at time $t-1$ a hole is present in station u. Suppose that the stations $u+1, \ldots, v-1$ have trains which do not move in step t, while the train at station v moves. Then we will say that the hole at u at time $t-1$ has jumped to v in step t. Similarly, if the trains at stations $0, 1, \ldots, s-1$ wait and the train at s moves, we consider that to be a jump of an external hole from station -1 to station s.

Note that in a single step, a hole can jump across any number of links, while a train can only move across one link (to an adjacent station). However, hole-jumps and train-movements cannot overlap, as delineated in Lemma 2, which follows from the above definitions.

▶ **Lemma 2.** *If two holes jump in the same step, then the paths of those jumps must be link-disjoint, i.e., they do not share any link. If a train-movement occurs in the same step as a hole-jump, then their paths too must be link-disjoint.*

3.1 Chains induced by a schedule

In any given schedule, consider an external train p that enters the line by filling a hole $\overline{h}$. Define $\overline{h}$ as the predecessor of p. If $\overline{h}$ is created by the exit of another train $\overline{p}$, then define $\overline{p}$ as the predecessor of $\overline{h}$. This procedure will link every external train into an alternating sequence of trains and holes. Each such maximal sequence is called a *chain*.

The first element of a chain can either be an internal train or a pre-existing hole, both of which do not have predecessors. That train or hole is said to be the *beginning* of the chain. Let c be a chain and station s_0 be the initial position of its beginning. Let p_k be the last train of c, and s_k the entry station of p_k. Then the *span* of chain c is defined as the interval $[s_0, s_k]$. The last train p_k is said to be the *terminal train* of the chain, while all other trains of the chain are said to be its *non-terminal trains* (Figure 4). Every link within the span of a chain c is either run over by a non-terminal train of c, or jumped over by a hole of c. We say that the chain *crosses* all these links. Note that the links run over by the terminal train of a chain are, by definition, not crossed by the chain.

▶ **Definition 3.** *Congestion produced in a link*, say l, by a set $\mathcal{C}$ of chains is defined as the number of those chains in $\mathcal{C}$ which cross l. *Congestion of a set* of chains is the maximum congestion produced by the set in any link (including those upstream of station 0).

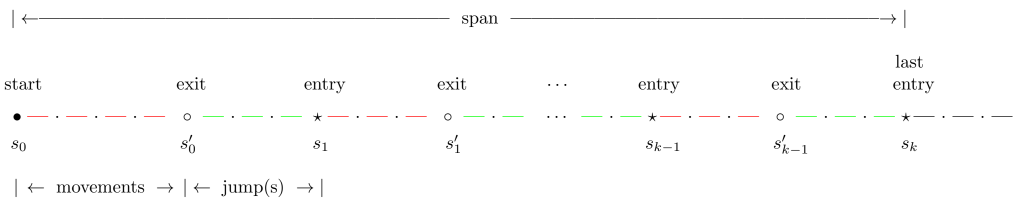

Figure 4 *Spatial view of a chain* $\langle p_0, h_0, p_1, h_1, \ldots, p_{k-1}, h_{k-1}, p_k \rangle$
Station s_0 is the origin ($\bullet$) of the internal train p_0 with which the chain begins. Stations $s_1, \ldots, s_{k-1}, s_k$ are the entry points ($\star$) of external trains $p_1, \ldots, p_{k-1}, p_k$. Stations $s'_0, s'_1, \ldots, s'_{k-1}$ are the destinations ($\circ$) of non-terminal trains $p_0, p_1, \ldots, p_{k-1}$, where the holes $h_0, h_1, \ldots, h_{k-1}$ get created when those trains exit. Links (–) crossed by train-movements are shown in red, while the links (–) crossed by hole-jumps are in green. Trains have not been explicitly depicted.

▶ **Definition 4.** We define the *age* of a chain c as the minimum possible time by which its terminal train, say p, can enter the line, i.e., the entry time t of p if it were the case that in every step before t, either a train of c moves on the line or a hole of c jumps and gets filled in.

In other words, the age of c equals the number of movements on the line to be made by its non-terminal trains, plus the number of its holes, since every hole h of a chain makes at least one jump – the one coinciding with the entry of the train that succeeds h in the chain. Note that *the age of a chain is a unit more than the sum of path-lengths of its non-terminal trains.*

▶ **Theorem 5.** *Let $\mathcal{S}$ be a schedule, $\mathcal{C}$ the set of chains induced by $\mathcal{S}$, and T the last-entry-time in $\mathcal{S}$. Then $\mathcal{C}$ has a congestion at most T, and every chain in $\mathcal{C}$ has an age at most T.*

Proof. At most one train can cross a link in a single step. When a hole jumps in a step t from a station s to another station s', all trains and holes at the stations between s and s' halt during step t. Hence, for any link l, at most one train or hole can move or jump across l in each step, which implies that at most T chains can cross l by time T. Since the terminal trains of all chains have entered the line by time T, no chain crosses any link afterwards. Thus, congestion of any link can not exceed T, implying that $\mathcal{C}$ has a congestion at most T.

For any chain, movements of its trains along the line must happen at distinct times. Furthermore, the hole-jumps must also happen at distinct times and every hole must make at least one jump. Hence, the last hole-jump, which coincides with the entry of the last train of the chain, cannot be made at a time smaller than the age of the chain. ◀

4 NP-hardness

For the decision version of Train Scheduling, the input is as given in Section 2 together with an integer T. We are required to decide if there is a schedule $\mathcal{S}$ with max-delay less than T.

▶ **Theorem 6.** *Train Scheduling is NP-hard.*

Proof. The reduction is from the Bin Packing problem [9], for which the input is (i) a finite set $U = \{X_1, \ldots, X_n\}$ of positive integers, (ii) an integer bin capacity B such that $B \geq X_i\ \forall i$, and (iii) a positive integer M. The required decision is whether a partition of U into M disjoint subsets $U_1, \ldots, U_M$ exists such that the sum of integers in every subset is at most B.

Given a Bin Packing instance, our Train Scheduling instance is as follows. We have a line with stations $0, \ldots, N$ along with an integer T, where $N = \alpha(3+(n+1)B)$, $T = \alpha(B+1)$, and $\alpha = (n+M)$. Stations $T+1, \ldots, T+M$ have holes $h_1, \ldots, h_M$; other stations have internal trains going to the last station N. For external trains, we first have T trains $p_1, \ldots, p_T$ at the outers of stations $0, \ldots, T-1$, each going to station N. We then have n external trains

$q_1, \ldots, q_n$, each q_i going a distance αX_i. These wait, in any order, at the outers of stations downstream of station $T+M$ such that their paths are disjoint, i.e., they share neither a link nor a station. Let D be the most downstream station among the destinations of q_is. We also have M external trains $r_1, \ldots, r_M$ at the outers of stations $D+1, \ldots, D+M$, each going to station N. It should be clear that for our chosen value of N, all these trains do fit within the network.[1] To complete the proof, we make the following three observations.

First, note that the time taken for the reduction is polynomial in n and B. Since Bin Packing is strongly NP-hard, we may assume its input to be in unary. Hence, the reduction runs in time polynomial in the size of the Bin Packing instance.

Second, suppose the Bin Packing instance has a solution $\{U_1, \ldots, U_M\}$. Then, for the Train Scheduling instance, we can build a schedule with max-delay less than T as follows. We partition the external trains into $T+M$ chains. For every $j \in \{1, \ldots, T\}$, we construct a chain c_j consisting of (i) the external hole at station $-j$, and (ii) the train p_j. For every $k \in \{1, \ldots, M\}$, we construct a chain c'_k consisting of (i) the hole h_k, (ii) $|U_k|$ trains (and the holes created by their exits) corresponding to the integers in U_k, and (iii) the train r_k. Note that the age of every c'_k is $1 + \sum_{X \in U_k} \alpha X \leq 1 + \alpha B$, while the age of every c_j is 1.

In each step $j \in \{1, \ldots, T\}$ we schedule the train p_j to enter the line. Note that the other $(n+M)$ entries further downstream the line do not conflict with these T entries. Therefore, to prove that none of the external trains gets delayed by more than $T-1$ steps, it suffices to show that the other entries can also be scheduled to take place by time T. In fact, we show that they can be made to take place by time $T-1$ as follows. In each step $k \in \{1, \ldots, M\}$ we schedule the entry of the first train of chain c'_k. Subsequently, in every step we prioritise entries over movements on the line. There can be at most $n+M$ steps in which the trains q_i and r_k enter the line. In other steps, for each c'_k, one of its non-terminal trains moves on the line unless its last train r_k has already entered; there can be at most $\alpha B - 1$ such steps. Thus, the total number of steps by the time every r_k has entered must be at most $n + M + \alpha B - 1 = T - 1$. Hence, the maximum delay for the external trains is $T-1$. This can be easily seen to hold also for the internal trains.[2]

Third, suppose the Train Scheduling instance has a schedule $\mathcal{S}$ with max-delay at most $T-1$. Then we can build a solution for the Bin Packing instance as follows. From Theorem 1, it follows that the last-entry-time in schedule $\mathcal{S}$ can at most be T. Consider the set $\mathcal{C}$ of chains induced by $\mathcal{S}$. Suppose there are more than $T+M$ chains in $\mathcal{C}$. Since all internal trains go till N, every chain must begin with a hole. Since there are only M internal holes, more than T chains must begin with external holes, implying a congestion more than T in the link $(-1, 0)$, which is a contradiction to Theorem 5. Hence, $\mathcal{C}$ has at most $T+M$ chains. Each p_j goes till the end and therefore must be the terminal train of its chain. Moreover, it cannot have any other train in its chain since no other train ends upstream of its entry station. Therefore, just p_js take T chains. Hence, the other $n+M$ external trains must be packed in the remaining chains, which then must be M in number since each r_k has to be the terminal train of one, say c'_k. From Theorem 5, age of every c'_k is at most $T = \alpha(B+1)$. That means the sum of path-lengths of all non-terminal trains of c'_k is less than $\alpha(B+1)$, which implies that the sum of integers corresponding to the non-terminal trains in c'_k is strictly less than $B+1$, i.e., at most B. Let U_k be the set of those integers. Thus, we get the required partition $\{U_1, \ldots, U_M\}$ of the input set U. ◀

---

[1] $D+M \leq \left(T + M + \sum_{i=1}^{n} \alpha X_i\right) + M \leq (\alpha(B+1) + M + n\alpha B) + M = \alpha + 2M + \alpha(n+1)B < N$

[2] The internal trains initially at stations $0, \ldots, T$ always move ahead during the first T steps while p_js are entering the line. None of the internal trains initially at stations $(T+M+1), \ldots, N$ halts after time $T-1$ by when all q_is and r_ks have entered the line. No train, internal or external, halts after time T.

5 A log factor approximation

We present a randomised approximation algorithm which builds a schedule achieving a last-entry-time $O(T^* \log N)$ with high probability (w.h.p.), where T^* is the optimal last-entry-time. Theorem 1 implies that it is also a log-factor approximation for minimising max-delay. The algorithm consists of two procedures :

1. The *partitioning* procedure takes as argument a target T. If T is a feasible last-entry-time then it returns a set of chains having $O(T)$ age and $O(T)$ congestion, otherwise it correctly declares T to be infeasible.
2. The *scheduling* procedure uses the set of chains returned by the partitioning procedure to generate a schedule having a last-entry-time $O(T \log N)$ w.h.p.

The overall algorithm runs in two stages. In the first stage, by performing a binary search on T in the range 1 through N, it finds the smallest value $\tilde{T}$ for which the partitioning procedure returns a set of chains. In the second stage, it invokes the scheduling procedure with the set of chains obtained for $\tilde{T}$ to get a schedule $\tilde{S}$. Since we know that no schedule is possible with last-entry-time less than $\tilde{T}$, the schedule $\tilde{S}$ – guaranteed to have last-entry-time $O(\tilde{T} \log N)$ – is a $\log N$ approximation of the optimal. Sections 5.1 and 5.2 give the details.

5.1 Partitioning

The partitioning procedure is given in Algorithm 1. It is called with a target time T. We use the term *short* or *long* for a train to denote whether its path-length is less than or at least T.

5.1.1 Analysis

We will compare the chains in $\hat{\mathcal{C}}$, as they get constructed by the procedure, with the chains in the set $\mathcal{C}^*$ induced by an optimal schedule, i.e., a schedule having a last-entry-time T. The comparison will show that at every station the set $\hat{\mathcal{C}}$ has more active chains – to which the external train (if any) can be added – than $\mathcal{C}^*$. Additionally, it will show that the active chains of $\hat{\mathcal{C}}$ have more capacity to accommodate external trains than the active chains of $\mathcal{C}^*$. This, in turn, will imply that if T is a feasible last-entry-time then for every external train the procedure has an active chain to add the train to, and hence it does not abort; rather, it runs to completion and returns the set of chains. In the following, we call a chain a *short chain* (*long chain*) if it ends with a short (long) train.

▶ **Definition 7.** We define the *weight* of a chain as the sum of the path-lengths of all short trains (terminal train, if short, as well as the non-terminal trains) in the chain.

▶ **Corollary 8.** *Given a target T for last-entry-time, let $\mathcal{C}^*$ be the set of chains induced by an optimal schedule, i.e., a schedule having a last-entry-time T. Then every chain in $\mathcal{C}^*$ has a weight at most $2T - 2$, and every non-terminal train in $\mathcal{C}^*$ is a short train.*

Before embarking on the analysis, we make some technical modifications to $\mathcal{C}^*$ as follows :

1. If $\mathcal{C}^*$ has $x < T$ chains beginning with external holes, then we additionally include $T - x$ *degenerate*[3] chains, each containing an external hole not already there in another chain.

[3] A chain consisting of just a hole or a short internal train (no external trains) is called a *degenerate* chain.

Algorithm 1 The partitioning procedure

1: Initialise set $\hat{\mathcal{C}}$ with $9T$ chains beginning with external holes at stations $-1, \ldots, -9T$.
2: Designate every chain in $\hat{\mathcal{C}}$ as *active*.
3: **for** each station $s = 0, 1, \ldots, N$ **do**
4:   **if** exists a short external train p at station s **then**
5:     **if** exist active chains with weight $< 5T$ and last train ending upstream of s **then**
6:       Add train p to any such chain c.
7:     **else**
8:       Declare T as infeasible and abort.
9:     **end if**
10:   **else if** exists a long external train p at station s **then**
11:     **if** exist active chains with last train ending upstream of s **then**
12:       Let c be any such chain with maximum weight.
13:       Add train p to chain c.
14:       *Terminate* chain c. {i.e., c is no longer active}
15:     **else**
16:       Declare T as infeasible and abort.
17:     **end if**
18:   **end if**
19:   **if** exists a hole h or a short internal train p at station s **then**
20:     **if** the number of active chains is $9T$ **then**
21:       Let $\bar{c}$ be any active chain with maximum weight.
22:       Terminate chain $\bar{c}$. {i.e., $\bar{c}$ is no longer active.}
23:     **end if**
24:     Add to $\hat{\mathcal{C}}$ a new chain c' beginning with h or p.
25:     Designate c' as active.
26:   **end if**
27: **end for**
28: Return $\hat{\mathcal{C}}$.

2. For every station $s \in \{0, \ldots, N\}$ having a hole h or a short internal train p, if $\mathcal{C}^*$ does not contain any chain beginning at s, then we add a degenerate chain containing h or p.
3. We extend the spans of short chains, without increasing the congestion of $\mathcal{C}^*$ beyond T, as follows. Let $[s, s']$ be the original span[4] of any chain $c \in \mathcal{C}^*$. Then the span of c is extended to $[s, s'']$, where s'' is as follows. If c is long, then $s'' = s'$; otherwise, s'' is the maximally downstream station from s' such that the congestion of $\mathcal{C}^*$ does not exceed T.

A chain $c \in \mathcal{C}^*$ that has an extended span $[s, s'']$ is said to *begin* at station s, be *active* at all stations and on all links in the open interval (s, s''), and be *terminated* at station s''. We say that $\mathcal{C}^*$ has been *maximally extended* by making these modifications. Note that this does not change the trains, holes, age, and weight of any chain already in $\mathcal{C}^*$.

▶ **Theorem 9.** *If the specified target time T is feasible, then the partitioning procedure completes successfully. Moreover, the set $\hat{\mathcal{C}}$ of chains it returns has a congestion at most $9T$, and each chain in the set has an age less than $6T$.*

Proof. First note that whenever the set $\hat{\mathcal{C}}$ has $9T$ active chains, the procedure terminates an active chain (Algorithm 1, line 20) before adding a new one (line 24). This implies that no

[4] The original span of a degenerate chain beginning at station s is taken to be $[s, s]$.

more than $9T$ chains are active on any link, i.e., the congestion of $\hat{\mathcal{C}}$ is at most $9T$. Moreover, once the weight of a chain exceeds $5T$, no more short trains are added to it. Hence, the weight of any chain can at most be $(5T-1)+(T-1)=6T-2$, and its age at most $6T-1<6T$.

Now suppose that T is a feasible last-entry-time. Then there must exist an *optimal schedule* $\mathcal{S}^*$ achieving it. Let $\mathcal{C}^*$ be the maximally extended set of chains induced by $\mathcal{S}^*$. For any set $\mathcal{C}$ of chains – particularly for $\mathcal{C} \in \{\hat{\mathcal{C}}, \mathcal{C}^*\}$ – let $X(\mathcal{C}, s)$ denote the number of chains active on the link $(s-1, s)$, and let $W(\mathcal{C}, s)$ denote *the total weight of active chains before* s, i.e., the sum of path-lengths of all those short trains that originate upstream of s and belong to the chains active on $(s-1, s)$. We will prove that the following hold at each station $s \in \{0, 1, \ldots N\}$:

Invariant I. $\Delta X(s) := X(\hat{\mathcal{C}}, s) - X(\mathcal{C}^*, s) = 8T$

Invariant II. $\Delta W(s) := W(\hat{\mathcal{C}}, s) - W(\mathcal{C}^*, s) < 26\,T^2$

The two invariants will imply that $\hat{\mathcal{C}}$ has an active chain to which the external train (if any) at s can be added, and hence the procedure does not abort (line 8 or 16).

The proof is by induction on stations. At station 0, the invariants clearly hold. Suppose they hold at stations $0, \ldots, s$. Then, to prove them at station $s+1$, we consider all the cases :

1. *No external train, but a long internal train p at s*

   In $\hat{\mathcal{C}}$ as well as $\mathcal{C}^*$, neither any train is added nor any chain begins at s. In $\hat{\mathcal{C}}$, by construction, no chain is terminated at s, implying $X(\hat{\mathcal{C}}, s{+}1) = X(\hat{\mathcal{C}}, s)$ and $W(\hat{\mathcal{C}}, s{+}1) = W(\hat{\mathcal{C}}, s)$. In $\mathcal{C}^*$, since $X(\mathcal{C}^*, s) \leq T$ and no chain begins at s, termination of a chain at s would mean that its extended span is not maximal – a contradiction to that $\mathcal{C}^*$ is maximally extended. Hence, in $\mathcal{C}^*$ as well no chain is terminated at s, that is, $X(\mathcal{C}^*, s{+}1) = X(\mathcal{C}^*, s)$ and $W(\mathcal{C}^*, s+1) = W(\mathcal{C}^*, s)$. Clearly, both the invariants hold at $s+1$.

2. *No external train, but a hole h or a short internal train p at s*

   In $\hat{\mathcal{C}}$ as well as $\mathcal{C}^*$, a chain begins at s. If $X(\hat{\mathcal{C}}, s) = 9T$, then from Invariant I, $X(\mathcal{C}^*, s) = T$, and hence a chain is terminated at s in each set (implied for $\hat{\mathcal{C}}$ by construction, and for $\mathcal{C}^*$ by congestion $\leq T$). Otherwise, no chain is terminated in either (by construction of $\hat{\mathcal{C}}$, and by $\mathcal{C}^*$ being maximally extended). Both ways, Invariant I holds at $s+1$.

   For Invariant II, we first note that in each of the two sets, the chain beginning at s with h or p contributes the same additional weight – 0 (if it is h) or some $\ell < T$ (if it is p) – to the total weight of active chains before $s+1$. If no chain is terminated at s, then Invariant II clearly holds at $s+1$. Otherwise, we need to consider two subcases :

   a. Suppose in $\hat{\mathcal{C}}$, the chain terminated at s has weight $2T$ or more, then $\Delta W(s+1)$ can only be smaller than $\Delta W(s)$, since in $\mathcal{C}^*$ the weight of the chain terminated at s can at most be $2T-2$. Therefore, Invariant II holds at $s+1$.

   b. Suppose in $\hat{\mathcal{C}}$, the chain terminated at s has weight less than $2T$. Then, since it has maximum weight among all chains active on $(s-1, s)$, each of the other active chains must also have a weight less than $2T$. The same chains are also active on $(s, s{+}1)$ with same weights before $s+1$ as before s. The only additional chain active on $(s, s+1)$ is the one beginning at s and having weight 0 or $\ell < T$. Then $X(\hat{\mathcal{C}}, s+1) \leq 9T$ implies $W(\hat{\mathcal{C}}, s+1) < 18\,T^2$. Since $W(\mathcal{C}^*, s+1) \geq 0$, Invariant II holds at $s+1$.

3. *A short external train p', and a long internal train p at s*

   Since in $\mathcal{C}^*$ the train p' must belong to some chain active on $(s-1, s)$, $X(\mathcal{C}^*, s) \geq 1$. Then Invariant I implies $X(\hat{\mathcal{C}}, s) \geq 8T+1$. Moreover, $X(\mathcal{C}^*, s) \leq T$ since the congestion of $\mathcal{C}^*$ is at most T, and from Corollary 8 the weight of any chain in $\mathcal{C}^*$ is at most $2T-2$. Hence, $W(\mathcal{C}^*, s) < 2T^2$, and by Invariant II, $W(\hat{\mathcal{C}}, s) < 26\,T^2 + 2T^2 = 28\,T^2$. In $\hat{\mathcal{C}}$, out of all chains active on $(s-1,s)$, only less than $2T$ may end at or downstream of s.[5] Thus, more than $6T$ active chains must end upstream of s. Then for the average weight, say $\overline{W}$, of upstream ending active chains, we have $\overline{W} < \frac{28\,T}{6} < 5T$. Since the average weight before s of the upstream ending active chains is less than $5T$, at least one of these chains must have weight less than $5T$, that is, p' can be added to it. Hence, the partitioning procedure will evaluate the condition in line 5 as true, and will not abort.

Train p, being a long internal train, does not belong to any chain, i.e., no chain begins or is terminated at s either in $\hat{\mathcal{C}}$ or in $\mathcal{C}^*$. Therefore, $X(\hat{\mathcal{C}}, s+1) = X(\hat{\mathcal{C}}, s)$, $X(\mathcal{C}^*, s+1) = X(\mathcal{C}^*, s)$, and $\Delta X(s+1) = \Delta X(s)$. Moreover, addition of p' increments the total weight of active chains by the same amount in both the sets, and hence does not change the difference, i.e., $\Delta W(s+1) = \Delta W(s)$. Thus, both the invariants hold at $s+1$.

4. *A short external train* p'*, and a hole* h *or a short internal train* p *at* s

   By the same argument as in case 3, the partitioning procedure will not abort; rather p' will get added to an active chain in $\hat{\mathcal{C}}$, incrementing the total weight of active chains by the same amount as in $\mathcal{C}^*$. Then, by the argument of case 2, the invariants hold at $s+1$.

5. *A long external train* p'*, and a long internal train* p *at* s

   By the same argument as in the earlier part of case 3, in $\hat{\mathcal{C}}$ the number of active chains ending upstream of s is more than $6T$. Therefore, the partitioning procedure will evaluate the condition in line 11 as true, and will not abort; rather it will add p' to a chain, say c, which in the current case will be terminated at s. Since one chain is terminated at s out of the $X(\hat{\mathcal{C}}, s)$ chains active on $(s-1,s)$ in $\hat{\mathcal{C}}$, $X(\hat{\mathcal{C}}, s+1) = X(\hat{\mathcal{C}}, s) - 1$. In $\mathcal{C}^*$ too, a chain (the one containing p') is terminated at s, i.e., $X(\mathcal{C}^*, s+1) = X(\mathcal{C}^*, s) - 1$. Hence, Invariant I holds at $s+1$. For Invariant II, we need to consider two subcases :

   a. Suppose in $\hat{\mathcal{C}}$, the chain terminated at s has weight $2T$ or more. Then, by the same argument as in case 2a, Invariant II holds at $s+1$.

   b. Suppose in $\hat{\mathcal{C}}$, the chain terminated at s has weight less than $2T$. Then each of the other upstream ending active chains too must have a weight less than $2T$, and we know that each of the (less than $2T$) downstream ending active chains has weight less than $6T$. The same chains are also active on $(s, s+1)$ with same weights before $s+1$ as before s. Then $X(\hat{\mathcal{C}}, s+1) \leq 9T$ implies $W(\hat{\mathcal{C}}, s+1) < 7T \cdot 2T + 2T \cdot 6T = 26\,T^2$. Since $W(\mathcal{C}^*, s+1) \geq 0$, Invariant II holds at $s+1$.

6. *A long external train* p'*, and a hole* h *or a short internal train* p *at* s

   In each of the sets $\hat{\mathcal{C}}$ and $\mathcal{C}^*$, p' is added to a chain which is then terminated at s, and a chain begins at s with h or p. For $\hat{\mathcal{C}}$ by construction, and for $\mathcal{C}^*$ being maximally extended, no other chain is terminated at s. Therefore, $X(\hat{\mathcal{C}}, s+1) = X(\hat{\mathcal{C}}, s)$ and $X(\mathcal{C}^*, s+1) = X(\mathcal{C}^*, s)$, implying that Invariant I holds at $s+1$. For the change in total weights due to the termination of a chain, the same arguments hold as in the cases 5a and 5b. Moreover, the chain that begins at s contributes to the total weight of active chains before $(s+1)$ by the same amount in $\hat{\mathcal{C}}$ as in $\mathcal{C}^*$. Hence, Invariant II also holds at $s+1$. ◀

5.2 Scheduling

If T is a feasible last-entry-time, then a chain set $\hat{\mathcal{C}}$ is returned by the partitioning procedure, and has a congestion $O(T)$ and maximum age $O(T)$. Clearly, the age and congestion are

[5] The last train $p^\dagger$ in such a chain c ends at or downstream of s, and has a path-length less than T since if $p^\dagger$ were long then c would already be terminated before s. Therefore, $p^\dagger$ must originate at one of the $T-1$ nearest stations before s, each of which has at most two trains – one internal and one external.

lower bounds on the last-entry-time with which the chains can be scheduled. However, to schedule them with last-entry-time better than $\Theta(T^2)$, the train-movements and hole-jumps of different chains need to be effectively pipelined. This is the goal of our scheduling procedure.

The procedure first splits the original big problem of scheduling the chains into several small scheduling problems. For this, it assigns a random initial rank $O(T)$ independently to every chain, and then successively incremental ranks to the holes and the movements of non-terminal trains of the chain. Since the total number of train-movements and holes in any chain is $O(T)$, the maximum value of rank assigned is $O(T)$. The original big problem has thus been broken down to as many small problems as the total number $\Gamma = O(T)$ of distinct ranks. The i^{th} small problem consists of the train-movements and holes of rank i, and the goal is to make every hole jump over its entire *extent* (Section 5.2.1) as well as to perform all the train-movements. Each small problem involves at most one hole or train-movement from every chain, and has a congestion $O(\log N)$ w.h.p. Note that the idea of using random delays in order to achieve effective pipelining is not new – it has been used in many previous works, e.g. [14] and [13].

Next, the procedure solves each small problem by using interval graph colouring to partition its set of train-movements and holes into $O(\log N)$ subsets. The colouring ensures that the extents of hole-jumps and train-movements in each subset are mutually disjoint, so that they can be scheduled to take place in a single step.

Thus, the procedure consists of two subroutines – (i) the *ranking subroutine* which assigns the ranks, and (ii) the *scheduling subroutine* which builds the schedule as a sequence of several phases, the i^{th} phase consisting of hole-jumps and train-movements of rank i. The number of distinct values of the ranks is $O(T)$, as we prove in Theorem 10. The number of steps in every phase is $O(\log N)$ w.h.p., as proved in Theorem 11. Hence, the schedule achieves a last-entry-time $O(T \log N)$ w.h.p.

5.2.1 The ranking subroutine

First, to every chain $c \in \hat{\mathcal{C}}$, independently assign a random initial rank $\gamma(c)$ from the range $\{1, 2, \ldots T\}$. Then, to every entry and movement in chain c (except movements following the entry of the terminal train, say $p^\dagger$, of c), assign a rank equal to the sum of $\gamma(c)$ and the number of previous movements and entries in the chain. Let $\gamma(c, \dagger)$ denote the rank of the last entry, that of the terminal train $p^\dagger$, in chain c. Let Γ be the maximum among all ranks.

For every link $(s-1, s)$ on the path of a train p on the line, the *extent of movement* of p across that link is defined as the interval $(s-1, s)$. For every external train p' that enters a station s' filling a hole h either pre-existing or created at some station $s < s'$, the *extent of entry* of p' is defined as (s, s'); the hole h is also said to have the same extent.

5.2.2 The scheduling subroutine

For each $i \in \{1, \ldots, \Gamma\}$, in i^{th} phase, all movements and entries having rank i are scheduled. Since those with overlapping extents cannot be scheduled in same step, a minimal interval colouring is first computed for the set of all extents with rank i. Movements and entries (only those not already scheduled for an earlier step as *incidental movements and entries*) whose extents have colour j are scheduled for the j^{th} step of the i^{th} phase; interval colouring ensures that they do not conflict. Thus, if K_i is the number of colours used in the colouring, then the i^{th} phase has K_i steps. Hence, last-entry-time of the schedule is at most $\sum_{i=1}^{\Gamma} K_i$. All external trains have entered the line by the end of the Γ^{th} phase. Therefore, after that, all trains not already vanished are trivially scheduled to move non-stop to their destinations.

Note that for the j^{th} step of i^{th} phase, along with the *designated* movements and entries (i.e., the ones having rank i and colour j), those movements (with higher value of rank or colour) are also scheduled which do not conflict with the designated ones. Similarly, those non-conflicting entries (with higher rank or colour) also take place in the same step whose holes-to-be-filled happen to move to the respective entry stations in this step. These additional movements and entries are what we earlier qualified as *incidental*.

5.2.3 Analysis

▶ **Theorem 10.** $\Gamma < 7T$

Proof. By definition, rank $\gamma(c, \dagger)$ assigned to the last entry of a chain c is one unit less than the sum of the age of c and its initial rank $\gamma(c) \leq T$. From Theorem 9, the age of every chain in $\hat{\mathcal{C}}$ is less than $6T$. Hence, $\gamma(c, \dagger) < 7T$ for every chain c, and $\Gamma = \max_c \gamma(c, \dagger) < 7T$. ◀

▶ **Theorem 11.** *For each $i \in \{1, \ldots, \Gamma\}$, number of steps in the i^{th} phase is $O(\log N)$ w.h.p.*

Proof. Let $\hat{\mathcal{C}}$ be the set of chains returned by the partitioning procedure for target time T. Let $r \leq \Gamma$ be any rank, and l be any link. From Theorem 9, the congestion produced by $\hat{\mathcal{C}}$ in l is at most $9T$. For each of at most $9T$ chains which cross l, extent of exactly one movement or entry of the chain includes l. Let E_l be the set of all extents which include l. Then $|E_l| \leq 9T$, and the extents in E_l belong to distinct chains.

For every extent $\epsilon \in E_l$, let X_ϵ be a binary random variable which takes a value 1 if the extent ϵ is assigned the rank r, and a value 0 otherwise. Note that the value of X_ϵ depends entirely on the random initial rank $\gamma(c_\epsilon)$ assigned to the chain, say c_ϵ, to which the extent ϵ belongs. Moreover, recall that the random rank is assigned to c_ϵ independently of other chains. Then, since the extents in E_l belong to different chains, X_ϵ are independent Bernoulli random variables. Furthermore, for any extent ϵ, at most one (if any) out of T equally probable values for $\gamma(c_\epsilon)$ can lead to rank r be assigned to ϵ, i.e., $P[X_\epsilon = 1] \leq \frac{1}{T}$.

Let $X := \sum_{\epsilon \in E_l} X_\epsilon$, i.e., X counts the extents which include l and have rank r. Then $\mu := \mathbb{E}[X] \leq \frac{|E_l|}{T} \leq 9$. Applying the Chernoff bound $P[X \geq \lambda] \leq \left(\frac{\lambda}{\mu e}\right)^{-\lambda}$ $\forall \lambda \geq 0$, we have :
$P[X \geq k \lg N] \leq \left(\frac{k \lg N}{9e}\right)^{-k \lg N} = \left(\frac{9e}{k \lg N}\right)^{k \lg N} \leq \frac{1}{4^{k \lg N}} = N^{-2k} \quad \forall k \geq 9e, \forall N \geq 16$
Thus, the probability that l is included in more than $k \lg N$ extents with rank r is less than N^{-2k}. Since there are only N links and only Γ different values for rank, and since $\Gamma < 7T$ from Theorem 10, the probability that any link is included in more than $k \lg N$ extents having same rank is less than $\frac{N \cdot 7T}{N^{2k}} < N^{-k}$. That is, the probability that no more than $k \lg N$ extents of same rank include a common link is greater than $(1 - N^{-k})$. Therefore, with a probability greater than $(1 - N^{-k})$, for every $i \in \{1, \ldots, \Gamma\}$, the number K_i of colours required in the minimal colouring of the extents with rank i is at most $k \lg N$, and hence the number of steps in the i^{th} phase is at most $k \lg N$. ◀

6 Conclusion

The most important open question is whether computing a constant factor approximation for the Train Scheduling problem is NP-hard. Natural generalisations – like multiple platforms at stations, unequal lengths of links, different speeds of trains, relative priorities of trains, and multiple parallel links between stations – should also be studied to bring the model closer to real-life problems.

References

1 V. Cacchiani, D. Huisman, M. Kidd, L. Kroon, P. Toth, L. Veelenturf, and J. Wagenaar. An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, 63:15–37, 2014. doi:10.1016/j.trb.2014.01.009.

2 X. Cai, C.J. Goh, and Alistair I. Mees. Greedy heuristics for rapid scheduling of trains on a single track. *IIE Transactions*, 30(5):481–493, May 1998. doi:10.1023/A:1007551424010.

3 Gabrio Caimi, Fabián A. Chudak, Martin Fuchsberger, Marco Laumanns, and Rico Zenklusen. A new resource-constrained multicommodity flow model for conflict-free train routing and scheduling. *Transportation Science*, 45(2):212–227, 2011. doi:10.1287/trsc.1100.0349.

4 Te-Wei Chiang, Hai-Yen Hau, Hwan-Ming Chiang, Su-Yun Ko, and Chao-Ho Hsieh. Knowledge-based system for railway scheduling. *Data Knowl. Eng.*, 27(3):289–312, 1998. doi:10.1016/S0169-023X(97)00040-2.

5 Andrea E. F. Clementi and Miriam Di Ianni. Optimum schedule problems in store and forward networks. In *Proceedings IEEE INFOCOM '94, The Conference on Computer Communications, Thirteenth Annual Joint Conference of the IEEE Computer and Communications Societies, Networking for Global Communications, Toronto, Ontario, Canada, June 12-16, 1994*, pages 1336–1343. IEEE, 1994. doi:10.1109/INFCOM.1994.337560.

6 Andrea D'Ariano. *Improving real-time train dispatching: Models, algorithms and applications*. Doctoral thesis, TRAIL Research School, Deft, The Netherlands, 2008.

7 Wei Fang, Shengxiang Yang, and Xin Yao. A survey on problem models and solution approaches to rescheduling in railway networks. *IEEE Trans. Intelligent Transportation Systems*, 16(6):2997–3016, 2015. doi:10.1109/TITS.2015.2446985.

8 Holger Flier, Matús Mihalák, Anita Schöbel, Peter Widmayer, and Anna Zych. Vertex disjoint paths for dispatching in railways. In Thomas Erlebach and Marco E. Lübbecke, editors, *ATMOS 2010 - 10th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems, Liverpool, United Kingdom, September 6-10, 2010*, volume 14 of *OASICS*, pages 61–73. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2010. doi:10.4230/OASIcs.ATMOS.2010.61.

9 Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

10 J. Harbering, A. Ranade, and M. Schmidt. Single track train scheduling. In Z. Hanzalek, G. Kendall, B. McCollum, and P. Sucha, editors, *In proceedings of the 7th Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA 2015), 25 - 28 Aug 2015, Prague, Czech Republic*, pages 102–117, 2015.

11 R. V. Iyer and S. Ghosh. Daryn-a distributed decision-making algorithm for railway networks: modeling and simulation. *IEEE Transactions on Vehicular Technology*, 44(1):180–191, February 1995. doi:10.1109/25.350284.

12 Johanna Törnquist Krasemann. Design of an effective algorithm for fast response to the re-scheduling of railway traffic during disturbances. *Transportation Research Part C: Emerging Technologies*, 20(1):62–78, 2012. Special issue on Optimization in Public Transport+ISTT2011. doi:10.1016/j.trc.2010.12.004.

13 Frank Thomson Leighton, Bruce M. Maggs, Abhiram G. Ranade, and Satish Rao. Randomized routing and sorting on fixed-connection networks. *J. Algorithms*, 17(1):157–205, 1994. doi:10.1006/jagm.1994.1030.

14 Frank Thomson Leighton, Bruce M. Maggs, and Satish Rao. Packet routing and job-shop scheduling in O(congestion + dilation) steps. *Combinatorica*, 14(2):167–186, 1994. doi:10.1007/BF01215349.

15 Frank Thomson Leighton, Bruce M. Maggs, and Andréa W. Richa. Fast algorithms for finding o(congestion + dilation) packet routing schedules. *Combinatorica*, 19(3):375–401, 1999. doi:10.1007/s004930050061.

16 Joseph Y.-T. Leung, Tommy W. Tam, and Gilbert H. Young. On-line routing of real-time messages. *J. Parallel Distrib. Comput.*, 34(2):211–217, 1996. doi:10.1006/jpdc.1996.0057.

17 Carlo Mannino and Alessandro Mascis. Optimal real-time traffic control in metro stations. *Operations Research*, 57(4):1026–1039, 2009. doi:10.1287/opre.1080.0642.

18 Alessandro Mascis and Dario Pacciarelli. Job-shop scheduling with blocking and no-wait constraints. *European Journal of Operational Research*, 143(3):498–517, 2002. doi:10.1016/S0377-2217(01)00338-1.

19 Sundaravalli Narayanaswami and Narayan Rangaraj. Scheduling and rescheduling of railway operations: A review and expository analysis. *Technology Operation Management*, 2(2):102–122, December 2011. doi:10.1007/s13727-012-0006-x.

20 Britta Peis, Martin Skutella, and Andreas Wiese. Packet routing: Complexity and algorithms. In Evripidis Bampis and Klaus Jansen, editors, *Approximation and Online Algorithms, 7th International Workshop, WAOA 2009, Copenhagen, Denmark, September 10-11, 2009. Revised Papers*, volume 5893 of *Lecture Notes in Computer Science*, pages 217–228. Springer, 2009. doi:10.1007/978-3-642-12450-1_20.

21 Paola Pellegrini and Joaquin Rodriguez. Single european sky and single european railway area: A system level analysis of air and rail transportation. *Transportation Research Part A: Policy and Practice*, 57:64–86, 2013. doi:10.1016/j.tra.2013.09.004.

22 Thomas Rothvoß. A simpler proof for o(congestion + dilation) packet routing. *CoRR*, abs/1206.3718, 2012. arXiv:1206.3718.

23 Ismail Sahin. Railway traffic control and train scheduling based oninter-train conflict management. *Transportation Research Part B: Methodological*, 33(7):511–534, September 1999. doi:10.1016/S0191-2615(99)00004-1.

24 Christian Scheideler. Universal routing strategies for interconnection networks. In G. Goos, J. Hartmanis, and J. van Leeuwen, editors, *Lecture Notes in Computer Science 1390*, pages 57–67. W. H. Freeman & Co., New York, NY, USA, 1998.