Freight Train Scheduling on a Single Line Corridor

Amirah Rahman[1] and Gary Froyland[2]

[1]*School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM Penang, Malaysia*
[2]*School of Mathematics and Statistics, University of New South Wales, Sydney 2052 NSW, Australia*

Abstract. Most of the world's freight rail network consists of single track lines with sidings where interactions between trains occur (meet, pass). The goal is to generate a schedule where the trains spend as little time as possible traversing between loading and unloading stations while ensuring that the interactions happen safely. Currently, scheduling is a manual operation performed by train controllers or dispatchers. The complexity of the problem makes it a demanding and highly time consuming task. We present an integer program formulation and results obtained from numerical experiments.

Keywords: optimisation, mixed integer programming, train scheduling
PACS:

INTRODUCTION

Train scheduling involves setting the departure and arrival times of all trains at all stations on their respective routes. Our interest is in scheduling freight trains on a bidirectional single-line track, also known as a *corridor*. The corridor is divided into segments, where each segment is bounded by two stations. Stations could either be sidings or endstations. Sidings are short stretches of double-lined track where two trains can meet and pass one another. Endstations are train yards situated at the ends of the corridor where trains can load or unload their goods. Trains transport goods from one end of the corridor to the other. Route information for each train is provided, but no initial feasible schedule is given. This means that our train scheduling problem is a *zero base problem*.

The freight train schedule to be determined must satisfy some operational and safety constraints. Operational constraints restrict the movement of each train separately on the track, while safety constraints monitor the interactions between trains and ensure that no conflicts arise. We consider two types of conflicts: (1) proximity conflicts, which occur when two trains are traversing the track too close to one another, and (2) collision conflicts, which occur when two trains simultaneously traverse the same segment, causing a collision.

Freight train scheduling is still mainly done manually by train controllers or dispatchers. The complexity of this problem makes this task highly time consuming. However, there have been studies that explore solving the zero base train scheduling problem using mathematical programming methods, such as [1,2,4]. The study by

[1] This research was carried out at the School of Mathematics and Statistics at the University of New South Wales.

Statistics and Operational Research International Conference (SORIC 2013)
AIP Conf. Proc. 1613, 159-168 (2014); doi: 10.1063/1.4894342

Zhou and Zhong (2007) [4] solves the problem of scheduling trains on a single line corridor, where each train only makes one origin-destination trip. The problem is formulated as a mixed integer program and uses branch-and-bound as the solution method, where two lower bound rules are introduced to reduce the search space.

The study by Castillo *et al.* (2009) [1] solves a similar problem to that in [4]. The authors use a 3-stage algorithm as their solution method. In the first stage, the bisection method was used to provide the relative travel times of all trains, while the second and third stages respectively allocate the trains and minimise the total dwell time of all trains. The study by Castillo *et al.* (2010) [2] improves upon the formulation in [1] by making changes to the objective function and conflict constraints as well as adding 3 constraint sets to the problem. The bisection method is also used as the solution method.

PROBLEM DESCRIPTION

We consider the problem of scheduling trains that make a fixed number of return trips on the corridor. On each return trip, a train begins its journey at one endstation, make its way to the other endstation and back again, loading and unloading at the endstations as it goes. Loaded trains are heavier, and therefore take more time to traverse the track. Each siding allows at most one pair of trains to meet and pass one another.

Trains are only allowed to meet and pass one another at stations to prevent collision conflicts from occurring. Both collision and proximity conflicts are dealt with using the following safety headways:

- Segment headway h.
  - Minimum time gap between a train leaving a segment and another train entering the same segment.
- Segment headway f.
  - Minimum time gap between a train departing a station and another train arriving the same station.
- Segment headway g.
  - Minimum time gap between two trains arriving at the same station.

All headways discussed above are depicted on a partial train track of 3 stations q, u and v in the following figure. We do not allow overtaking to occur. An example of each headway is shown using a pair of trains interacting on the track.

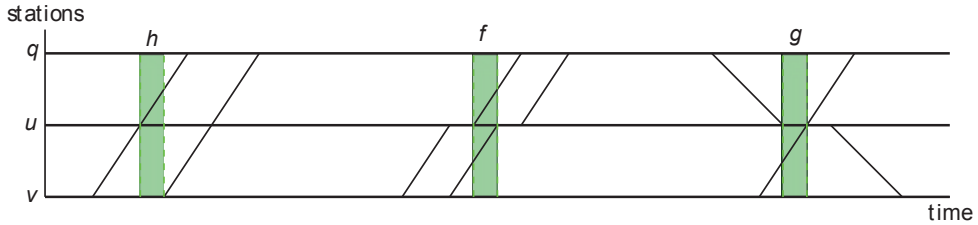

FIGURE 1. Headways h, f and g.

Our goal is to obtain, in as little time as possible, a schedule that minimises the arrival time of the last train at its last endstation.

Comparison With Prior Work

The studies mentioned in the previous section all share the similarity of having trains that only make one origin-destination trip for the whole duration under consideration. Our problem considers trains that make a set number of return trips before their journey can end. Similar to [1,2], we choose to base our formulation on the one put forward by [4]. We adapt the formulation in [4] to schedule trains that make multiple return trips between loading and unloading stations. We then extend the formulation with the intention of reducing the computation time of when solving the problem. We accomplish this by making certain observations that exploit the problem structure and express these observations in terms of constraints that we add to the adapted formulation.

MATHEMATICAL MODEL OF THE PROBLEM

We begin by stating the notation used in our formulation.

TABLE 1. Subscripts and Parameters

Symbol	Definition
$\mathcal{I}$	$=\{1,...,I\}$, the set of trains.
i	Train index.
$\mathcal{R}$	$=\{1,...,R\}$, the set of one-way trips the trains make.
r	Trip index.
$\mathcal{J}$	$=\{1,...,J\}$, the set of all segments in the network.
j	Segment index.
k	Segment sequence number in a route.
$\sigma(i,r,k)$	Segment index of the k^{th} segment in a route for train i on its r^{th} trip.
$\mathcal{U}$	$=\{1,...,U\}$, the set of all stations in the network.
u	Station index.
$\beta(i,r,k)$	Downstream station number of the k^{th} segment in a route for train i on its r^{th} trip.
$H(i)$	Number of segments in a route for train i.
$deptime(i)$	Earliest departure time of train i at its very first station.
$load(i,r)$	Load status of train i on trip r. =1 if train i on trip r is loaded. =0 otherwise.
$o(i,r,k)$	Orientation of train i on trip r at segment k. =1 if train i on trip r is heading towards a loading station. =0 if train i on trip r is heading towards an unloading station.
p_j^o	Traversing time for trains with orientation o at segment j.
$\underline{lo}$	Minimum required loading time at loading stations.

$\underline{ul}$	Minimum required unloading time at unloading stations.
f	Minimum headway between arrival and departure times of two consecutive trains (going in the same direction) at a station.
g	Minimum headway between arrival times of two consecutive trains at a station.
h	Minimum headway between arrival and departure times of two consecutive trains at a segment.
M	Sufficiently large constant.

TABLE 2. Decision Variables

Symbol	Definition
$s_{i,r,j}$	Start time for train i on trip r at segment j.
$e_{i,r,j}$	End time for train i on trip r at segment j.
$y_{i,r,i',r',j}$	=1 if train i on trip r enters segment j before train i' on trip r' enters the same segment. =0 otherwise.
$z_{i,r,i',r',u}$	=1 if train i on trip r arrives at station u before train i' on trip r' arrives at the same station. =0 otherwise.

Objective

The objective is to minimise the arrival time of the last train at its last endstation. That is,

$$\min\ (\max_i(e_{i,R,\sigma(i,R,H(i))})). \tag{1}$$

Adapted Constraints: Operational Restrictions

The following constraints are adapted from [4] for trains that make multiple trips before their journey ends. Trains must begin their journey no earlier than the given earliest departure time.

$$s_{i,1,\sigma(i,1,1)} \geq deptime(i), \qquad \forall i \in \mathcal{I}. \tag{2}$$

Trains must take up a specific amount of time to traverse the segments.

$$e_{i,r,\sigma(i,r,k)} \geq s_{i,r,\sigma(i,r,k)} + p^{load(i,r)}_{\sigma(i,r,k)}, \qquad \forall i \in \mathcal{I}, r \in \mathcal{R}, k \in \{1,...,H(i)\}. \tag{3}$$

Trains are allowed to dwell at all stations.

$$s_{i,r,\sigma(i,r,k)} \geq e_{i,r,\sigma(i,r,k-1)}, \qquad \forall i \in \mathcal{I}, r \in \mathcal{R}, k \in \{2,...,H(i)\}. \tag{4}$$

Trains must stop and load or unload at the endstations.

$$s_{i,r,\sigma(i,r,1)} \geq e_{i,r-1,\sigma(i,r-1,H(i))} + \underline{lo}, \qquad \forall i \in \mathcal{I}, r \in \{2,...,R\}, \text{where } load(i,r) = 1. \tag{5}$$

$$s_{i,r,\sigma(i,r,1)} \geq e_{i,r-1,\sigma(i,r-1,H(i))} + \underline{ul}, \qquad \forall i \in \mathcal{I}, r \in \{2,...,R\}, \text{where } load(i,r) = 0. \tag{6}$$

Adapted Constraints: Safety Restrictions

The following constraints are adapted from [4] for trains that make multiple trips before their journey ends. All pairs of trains must not violate any safety headways when traversing each segment and station to prevent the occurrence of proximity and collision conflicts. The constraint sets below ensure that safety headways h, f and g are respectively observed. The value of M is determined by solving the problem using a greedy heuristic and rounding up (to the nearest thousand) the objective value yielded by the heuristic.

$$\begin{aligned} s_{i,r,\sigma(i,r,k)} &\geq e_{i',r',\sigma(i',r',k')} + h - My_{i,r,i',r',j}, \\ s_{i',r',\sigma(i',r',k')} &\geq e_{i,r,\sigma(i,r,k)} + h - M(1-y_{i,r,i',r',j}), \end{aligned} \quad \begin{aligned} &\forall i \in \{1,...,I-1\}, i' \in \{i+1,...,I\}, \\ &r,r' \in \boldsymbol{\mathcal{R}}, j \in \boldsymbol{\mathcal{J}}, \\ &k \in \{1,...,H(i)\}, k' \in \{1,...,H(i')\}, \\ &\text{where } \sigma(i,r,k) = \sigma(i',r',k') = j. \end{aligned} \tag{7}$$

$$\begin{aligned} e_{i,r,\sigma(i,r,k)} &\geq s_{i',r',\sigma(i',r',k'+1)} + f - Mz_{i,r,i',r',u}, \\ s_{i',r',\sigma(i',r',k')} &\geq e_{i,r,\sigma(i,r,k+1)} + f - M(1-z_{i,r,i',r',u}), \end{aligned} \quad \begin{aligned} &\forall i \in \{1,...,I-1\}, i' \in \{i+1,...,I\}, \\ &r,r' \in \boldsymbol{\mathcal{R}}, u \in \boldsymbol{\mathcal{U}}, k \in \{1,...,H(i)-1\}, \\ &k' \in \{1,...,H(i')-1\}, \\ &\text{where } \beta(i,r,k) = \beta(i',r',k') = u. \end{aligned} \tag{8}$$

$$\begin{aligned} e_{i,r,\sigma(i,r,k)} &\geq e_{i',r',\sigma(i',r',k')} + g - Mz_{i,r,i',r',u}, \\ e_{i',r',\sigma(i',r',k')} &\geq e_{i,r,\sigma(i,r,k)} + g - M(1-z_{i,r,i',r',u}), \end{aligned} \quad \begin{aligned} &\forall i \in \{1,...,I-1\}, i' \in \{i+1,...,I\}, \\ &r,r' \in \boldsymbol{\mathcal{R}}, u \in \boldsymbol{\mathcal{U}}, \\ &k \in \{1,...,H(i)\}, k' \in \{1,...,H(i')\}, \\ &\text{where } \beta(i,r,k) = \beta(i',r',k') = u. \end{aligned} \tag{9}$$

Additional Constraints

The following constraints are added to the formulation with the intention of reducing the solve time and extend the constraints in [4]. This work appears in Rahman (2013) [3].

Consider a train pair i and i', traversing in the same direction, that share segment (respectively station) legs, where a segment (respectively station) leg is a set of consecutive segments (respectively stations) shared by the train pair. Figure 2 illustrates an example of two such trains journeying on those overlapping segments.

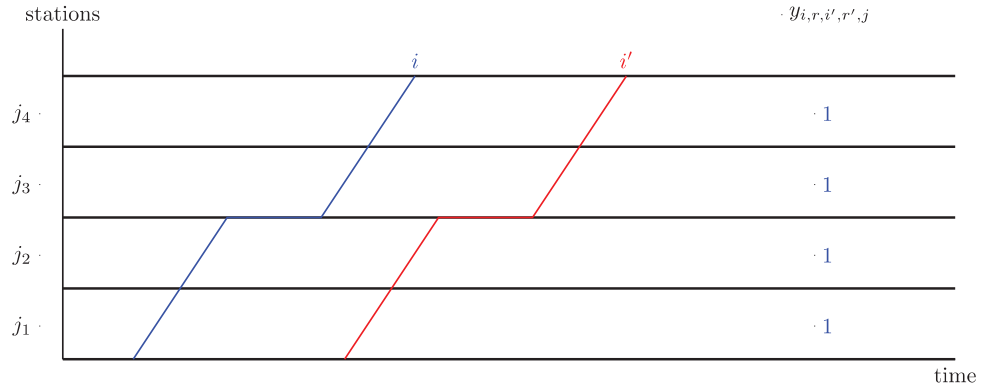

FIGURE 2. Precedence (same direction of travel)

As overtaking is not allowed on the track, the order in which the trains i and i' arrive at these shared segments (respectively stations) remains the same. This means that as train i traverses segment j_1 before train i', train i will also traverse segments j_2, j_3 and j_4 before train i' as well. That is, the binary y (respectively z) variables for the train pair at those segments (respectively stations) are the same. This is captured in the constraints below.

$$y_{i,r,i',r',j} = y_{i,r,i',r',fsj(l)}, \quad \begin{array}{l} \forall i \in \{1,...,I-1\}, i' \in \{i+1,...,I\}, r, r' \in \mathcal{R}, \\ l \in \{1,...,tspj(i,r,i',r')\}, j \in \{fsj(l)+1,...,lsj(l)\}, \\ \text{where } o(i,r,k) = o(i',r',k'), \\ fsj(l) = \text{first shared segment for } i \text{ and } i' \text{ on shared leg } l, \\ lsj(l) = \text{last shared segment for } i \text{ and } i' \text{ on shared leg } l, \\ tspj(i,r,i',r') = \text{no. of segment legs shared by } i \text{ and } i'. \end{array} \tag{10}$$

$$z_{i,r,i',r',u} = z_{i,r,i',r',fsu(l)}, \quad \begin{array}{l} \forall i \in \{1,...,I-1\}, i' \in \{i+1,...,I\}, r, r' \in \mathcal{R}, \\ l \in \{1,...,tspj(i,r,i',r')\}, u \in \{fsj(u)+1,...,lsj(u)\}, \\ \text{where } o(i,r,k) = o(i',r',k'), \\ fsu(l) = \text{first shared station for } i \text{ and } i' \text{ on shared leg } l, \\ lsu(l) = \text{last shared station for } i \text{ and } i' \text{ on shared leg } l, \\ tspu(i,r,i',r') = \text{no. of station legs shared by } i \text{ and } i'. \end{array} \tag{11}$$

EXPERIMENTS AND RESULTS

We test the performance of the formulation with the additional constraints (10) and (11) by performing numerical experiments that schedule 7 trains on a bidirectional single line corridor with 25 segments and 26 stations, where both endstations are able to load and unload goods. We use Cplex 12.4 as our solver.

Instances

We conduct our test over 72 instances, where each instance has a unique combination of the following:

- Formulation.
  - orig: Original (adapted) formulation.
  - add: Formulation with adapted constraints.
- Starting placements.
  - alternate: Trains are positioned such that both endstations are used as starting points for trains.
  - bunched: Trains are bunched up so that all trains begin their journey at the same endstation.
- Starting times.
  - time1: All trains begin at $t = 0$.
  - time2: Trains beginning their journey from the same endstation are given start times in 60 minute intervals, beginning with 0.
  - time3: Randomly generated, $t = \{324, 282, 150, 166, 176, 16, 333\}$.
- Return trips – 2,3 or 4 return trips.
- Cplex presolve – turn off or on.

The solve time of each instance is limited to 5400s.

Results

We measure performance in two ways:

- Quantity: the number of instances where at least one feasible solution is found within the time limit.
- Quality: how close the feasible solutions are to optimality.

We will first examine the performance of both the original formulation (orig) and the formulation with the added constraint set (add). Figure 3 below shows two cumulative graphs where the number of instances is plotted against the optimality gap.

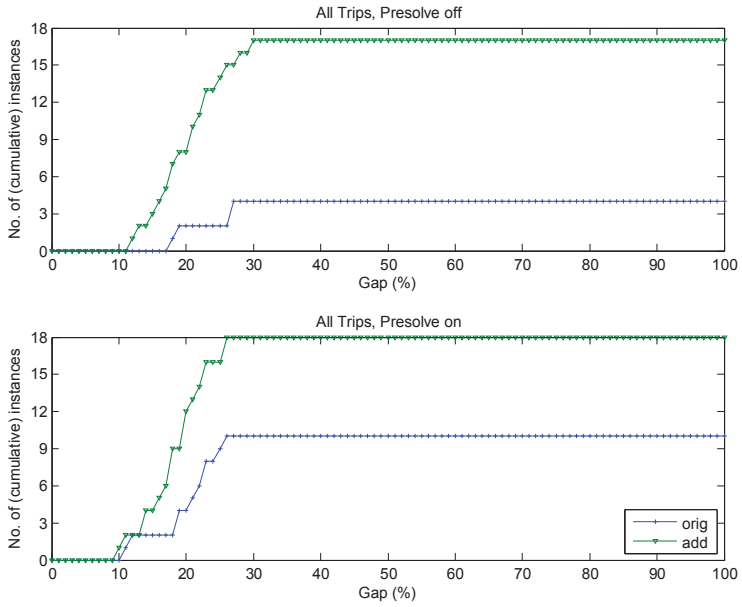

FIGURE 3. Cumulative graphs for all instances

Figure 3 (top) shows instances solved with presolve turned off, while the Figure 3 (bottom) shows instances solved with presolve turned on. The x-axis in both graphs represents the optimality gap, x, in percentage, while the y-axis represents the cumulative number of instances that are solved to within $\leq x\%$ of the optimal solution. Each line in both graphs represent instances that are solved using either (orig) or (add).

Both graphs clearly show that: (1) Most of the instances yielded feasible solutions within the given time limit, and (2) (add) vastly outperforms (orig). We can also see that turning on presolve greatly improved the performance of (orig), resulting in a high increase in the number of instances that yield feasible solutions in the given time limit. However, turning presolve on had less of a significant influence on (add). While (add) clearly improves upon (orig) in terms of quantity, the quality of solutions yielded by (add) are only slightly better than that yielded by (orig).

We now examine the performance of (orig) and (add) separately over instances that make 4, 6 and 8 trips respectively. Figure 4 below, which shows the cumulative graphs where trains make 4, 6 and 8 trips respectively, has a similar format to Figure 3.

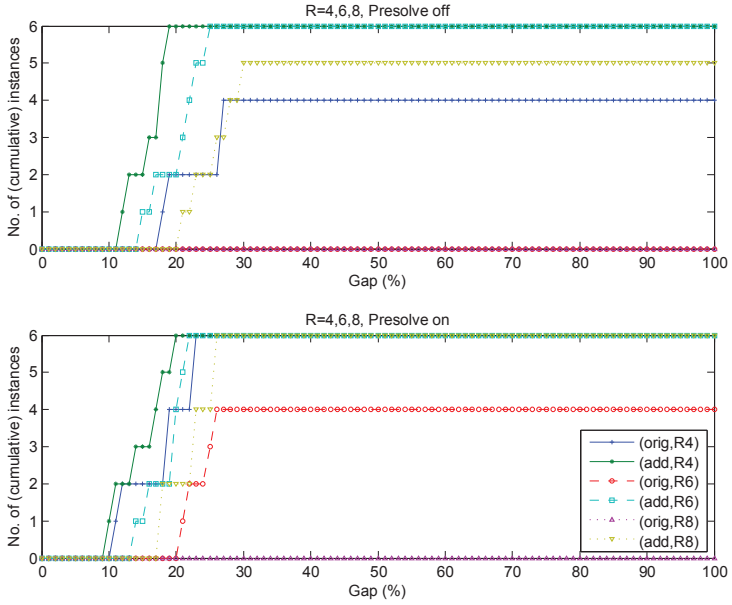

FIGURE 4. Cumulative graphs for instances where $R = 4, 6, 8$ respectively

In both graphs, the solid lines represent instances where $R = 4$, the dashed lines represent instances where $R = 6$ and the dotted lines represent instances where $R = 8$. For instances where $R = 4$, (add) performs slightly better than (orig) both in terms of quality and quantity. For instances where $R = 6$, (add) outperforms (orig) in both quality and quantity. For instances where $R = 8$, none of the instances using (orig) are able to find a feasible solution within the given time limit. While using (add), feasible solutions are found for 11 out of 12 instances. Notice that the larger the number of trips, the worse the performance of (orig) when compared to (add). Table 3 below, which shows the number of unsolved instances when using both formulations, illustrates this fact.

TABLE 3. Number of Unsolved Instances.

R	(orig)	(add)
4	2	0
6	8	0
8	12	1

CONCLUSION

Our results show that the formulation with the added constraint set vastly improves upon the original (adapted) formulation, especially for problems of larger size (higher number of trips, R). This is probably due to the fact that the added constraints exploit the following problem characteristic: All trains share the whole track with one another

and overtaking is not allowed on the track. Therefore, for any train pair on their respective trips traversing the track in the same direction, it suffices to determine the order in which the train pair traverses the whole track simply by determining the order on a single segment and station.

Further Study

Further study of the zero base train scheduling problem could include the following:

- Explore solving the problem on different track topology. For instance, track networks with junctions and multiple endstation.
- Tease out other problem characteristics that we can exploit and implement in the form of constraint sets.
- Develop a heuristic algorithm that could quickly solve the problem and provide a feasible solution to be used as a warm start when solving in Cplex.

ACKNOWLEDGMENTS

This work is part of a PhD project generously funded and supported by Universiti Sains Malaysia, the Ministry of Higher Education Malaysia, the School of Mathematics and Statistics at the University of New South Wales and the Australian Research Council Centre of Excellence for Mathematics and Statistics of Complex Systems (MASCOS).

REFERENCES

1. E. Castillo, I. Gallego, J. Ureña, and J. Coronado. "Timetabling optimization of a single railway track line with sensitivity analysis" in *TOP*, 17(2), 2009, pp. 256-287.
2. E. Castillo, I. Gallego, J. Ureña, and J. Coronado. "Timetabling optimization of a mixed double- and single- tracked railway network" in *Applied Mathematical Modelling*, 2010.
3. S.A.A.Rahman. "Freight train scheduling on a single line network", Ph.D. Thesis, University of New South Wales, 2013.
4. X Zhou and M. Zhong. "Single-track train timetabling with guaranteed optimality: Branch-and-bound algorithms with enhanced lower bounds" in *Transportation Research Part B: Methodological*, 41(3), 2007, pp. 320-341.