Train Shunting with Service Scheduling in A High-Speed Railway Depot

Xiaoming Xu
Maged M. Dessouky

July 21, 2022

Abstract

At high-speed railways, trains cover services during the day and are required to undergo maintenance at depots each night. A low-quality train schedule in the depot may result in delays in the availability of trains during the day which influences the reliability of the train timetables. Accordingly, this study examines the problem of train shunting with service scheduling in a depot where daily maintenance, cleaning operation, and safety operational requirements are considered. To cope with this complex problem, we first construct a two-layer time-space network in which each layer can only be used by trains traveling in the same direction. We then formulate the considered problem as a minimum-cost multi-commodity network flow model with incompatible arc sets and operational constraints. To solve the network flow problem, we present a Lagrangian relaxation heuristic. Finally, several computational experiments with practical data based on the Hefei-Nan depot and randomly generated data on trains’ arrival and departure times at the depot are conducted to confirm the effectiveness of our model and the efficiency of the proposed heuristics.

Keywords: Train shunting; maintenance; schedule; time-space network; Lagrangian relaxation

*School of Automotive and Transportation Engineering, Hefei University of Technology, Hefei, China; email: xmxu@hfut.edu.cn. Corresponding author.
†Daniel J. Epstein Department of Industrial and Systems Engineering, University of Southern California, Los Angeles, California 90089-0193, U.S.A.; email: maged@usc.edu
1 Introduction

To ensure safety and achieve high passenger satisfaction, trains (electric multiple units; EMUs) at China’s high-speed railways (CHSR) are required to undergo daily maintenance each night, in which some daily service tasks, such as train cleaning, maintenance inspections and repairs are carried out. This daily maintenance is performed in a depot which is usually near a station, as shown in Figure 1 (see, e.g., Haahr et al. 2017). Given a train timetable for the arriving and departing trains, the schedule of shunting trains in the depot needs to be determined such that train daily maintenance can be carried out at specific locations during these trains’ respective normal operating hours. Such a schedule plays an important role in railway operation because it can directly influence the reliability of the train timetable during the day. A low-quality train shunting schedule may result in delays in the availability of departing trains during the day, which in turn results in delays in the departures of planned train services. Even worse, low-quality shunting schedules may decrease the efficient use of depot resources, which may result in the rejection of some arriving trains to be checked and/or parked, thus leading to cancellations of some planned train services for the next day.

Following the definition of Freling et al. (2005) and Kroon et al. (2008), the process of parking train units, together with several related processes, is called *shunting*, and the corresponding planning problem is then called the Train Unit Shunting Problem (TUSP) (see, e.g., Lentink et al. 2006) which consists of parking, matching and routing, see, e.g., van den Broek et al. (2022). In the matching part, arriving train units are coupled to the required train compositions and assigned to departing trains in which the types of these coupled units are compatible. Since the maintenance resources, such as cleaning equipment and repair tracks in a depot, are limited, determining when and where service tasks for each train take place is a challenging task for operators. The routing part determines the route and corresponding timings of train units over the depot infrastructure such that safety operational requirements and track capacity constraints are satisfied. These are complex problems, especially in a dense depot for operators to determine manually and need computerized tools to help them solve these problems.

In this study, we examine a problem referred to as train shunting problem with service scheduling (TSPwSS, see, e.g., TUSPwSS in van den Broek et al. 2022) in the depots of CHSR consisting of train parking, routing and service scheduling where no matching decisions are expected to be determined, as matching decisions are

![Figure 1: Layout of a typical electric multiple unit depot.](image-url)
usually determined in rolling stock scheduling at CHSR (see, e.g., Zhong et al. 2019). The considered problem is NP-hard because it covers several NP-hard problems as special cases, such as the train routing problem (see, e.g., Caprara et al. 2011) and the train unit shunting problem (see, e.g., Jacobsen and Pisinger 2011) as well as the service task scheduling problem (see, e.g., van den Broek et al. 2022). We model this problem using an integer programming formulation based on a time-space network, and develop a Lagrangian relaxation heuristic to solve the considered problem. Several computational experiments are conducted with practical data based on the Hefei-Nan depot and randomly generated data on train arrival and departure times to confirm the efficiency and effectiveness of the developed heuristic.

The literature includes a great deal of research on train shunting problem. Blasum et al. (1999) proved the NP-completeness of the problem of dispatching trams in a depot with the objective of minimizing the number of shunting movements. More special cases of dispatching trams in a depot were addressed by Winter and Zimmermann (2000). Tomii et al. (1999) and Tomii and Zhou (2000) proposed a two-stage algorithm based on the genetic algorithm and a “program evaluation and review technique” for a simple version of TUSP in which a single train unit can park at a shunt track at each time point. Freling et al. (2005) introduced the problem of shunting passenger train units in a railway station. To cope with this complex problem, Freling et al. decomposed their problem into two smaller subproblems, including a matching problem and a track allocation (or parking) problem, where the former is solved by mixed integer program solver and the latter is solved using a column generation heuristic. Kroon et al. (2008) later presented a new model for TUSP that could solve the matching and parking subproblems in an integrated manner. Lentink et al. (2006) introduced the routing subproblem of TUSP and presented a four-step algorithm solution approach. Jacobsen and Pisinger (2011) studied the problem of shunting train units in a railway workshop area in which train unit coupling and decoupling were not considered. Considering the complexity of their problem, Jacobsen and Pisinger proposed three heuristic approaches. Haahr et al. (2017) presented several solution methods for TUSP, including a constraint programming formulation, a column generation approach, and a randomized greedy heuristic. Peer et al. (2018) studied a TUSP consisting of parking and matching problems under uncertainty, and developed a deep reinforcement learning solution method. Based on methods used in the container industry to stack containers, Beertuizen (2018) proposed various strategies to solve the parking and matching problems of TUSP, which results in two decision rules, including the “type based strategy” and the “in residence time strategy.” Moreover, van den Broek (2016; 2022) developed a local search approach for the train shunting and scheduling problem with the consideration of train matching, parking, and service tasks scheduling as well as train routing decisions. Interested readers are referred to Cordeau et al. (1998) and Lusby et al. (2011a) for comprehensive reviews of TUSPs.

Our problem is also related to the problem of routing or dispatching trains through railway stations. Caprara et al. (2007) performed a comprehensive survey of the train routing problem. The train routing problem has three versions. The easiest version requires that the planned train arrival and departure times are fixed and that the paths used by the trains are uniquely determined once the platforms have been selected, see, e.g.,
de Luca Cardillo and Mione (1998) and Billionnet (2003). A more complex version allows the planned arrival and departure times to be changed, whereas the paths traversed by the trains are uniquely determined by the choice of the platforms, see, e.g., Carey and Carville (2003). The more general version of the problem allows the planned train arrival and departure times to be changed, and the arrival and departure paths are not fixed, see, e.g., Zwaneveld et al. (1996; 2001), Caprara et al. (2011), Lusby et al. (2011b), Corman et al. (2009) and Caimi et al. (2011). As train routing problems at railway networks are becoming more complex, many fast heuristic approaches are particularly proposed to find near-optimal solutions. For example, Murali et al. (2016) presented a decision tool with an integer programming and a genetic algorithm for routing freight trains through complex networks. Samà et al. (2016) developed an ant colony optimization meta-heuristic for real-time train routing in railway networks. Samà et al. (2017) proposed a variable neighbourhood search for the real-time train scheduling and routing problem. Zhang et al. (2019) designed a decomposition scheme-based iterative algorithm for integrating train timetabling and track maintenance task scheduling.

Our problem falls into a daily rolling stock maintenance servicing work which is a part of the rolling stock maintenance problem. Similar to that at some European railways, such as Dutch railways (see, e.g., van Hövell 2022), the rolling stock maintenance logistics system in China also consists of transportation, (daily) servicing, heavy maintenance and component repair. Some representative works on rolling stock maintenance include Tönissen et al. (2019), Tönissen and Arts (2018; 2020), Canca and Barrena (2018), and Zomer et al. (2021), who focused on maintenance location decisions with different considerations, such as, unknown/uncertain train lines and fleet size, recovery costs of maintenance location decisions, allocation restrictions of the rolling stock types and maintenance timing of individual rolling stock units. Some studies combined maintenance scheduling with the assignment of rolling stock units to train trips such that maintenance constraints are satisfied, see, e.g., van Hövell (2022), Wagenaar and Kroon (2015) and Zomer et al. (2021).

Table 1 summarizes the detailed characteristics of some of the closely related literature. As can be seen in Table 1, our work differs from traditional studies of TUSPs in that our optimization model makes train parking, routing and service scheduling decisions simultaneously. Our model, however, does not include train unit coupling and decoupling decisions (see, e.g., matching decisions in Freling et al. 2005 and Kroon et al. 2008), but it includes minimum headway requirements at track joints, which are not covered in most models of TUSPs but are quite common in train platforming/routing models, see, e.g., Carey and Carville (2003) and Caprara et al. (2011). Our research topic bears some similarities to that of Jacobsen and Pisinger (2011) and van den Broek (2016; 2022) because our study also considers the assignment of tracks to trains (e.g., parking decisions) and the routing decisions of trains in the depot, whereas Jacobsen and Pisinger (2011) did not consider the minimum headway requirements at track joints, and van den Broek (2016; 2022) dealt with safety requirements at tracks by assigning buffer time to tasks rather than imposing a minimum headway restriction at track joints.

This paper therefore makes the following contributions to the literature. First, to the best of our knowledge, our study is the first to examine an integrated optimization approach to the train shunting with service scheduling in a high-speed railway depot, that considers both maintenance requirements and safety operational
Table 1: Comparison of some closely related studies.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Parking decisions</th>
<th>Matching decisions</th>
<th>Routing decisions</th>
<th>Service scheduling decisions</th>
<th>Safety at track (joints)</th>
<th>Solution approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beerthuizen (2018)</td>
<td>Considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>H</td>
</tr>
<tr>
<td>Freling et al. (2005)</td>
<td>Considered</td>
<td>Considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>MIP, CG</td>
</tr>
<tr>
<td>Haahr et al. (2017)</td>
<td>Considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>CP, MIP, H, (CG)</td>
</tr>
<tr>
<td>Jacobsen and Pisinger (2011)</td>
<td>Considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>H</td>
</tr>
<tr>
<td>Kroon et al. (2008)</td>
<td>Considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>MIP</td>
</tr>
<tr>
<td>Lentink et al. (2006)</td>
<td>Considered</td>
<td>Considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Considered (buffer time)</td>
<td>MIP, CG</td>
</tr>
<tr>
<td>Peer et al. (2018)</td>
<td>Considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Not considered</td>
<td>H</td>
</tr>
<tr>
<td>van den Broek (2016)</td>
<td>Considered</td>
<td>Considered</td>
<td>Considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>H</td>
</tr>
<tr>
<td>van den Broek et al. (2022)</td>
<td>Considered</td>
<td>Considered</td>
<td>Considered</td>
<td>Considered</td>
<td>Considered (buffer time)</td>
<td>H</td>
</tr>
<tr>
<td>Caprara et al. (2011)</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Considered (minimum headway)</td>
<td>ILP, B&amp;C&amp;P, H</td>
</tr>
<tr>
<td>Carey and Carville (2003)</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Considered (minimum headway)</td>
<td>H</td>
</tr>
<tr>
<td>Lusby et al. (2011b)</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Considered (block section)</td>
<td>ILP, B&amp;P</td>
</tr>
<tr>
<td>Zwaneveld et al. (2001)</td>
<td>Not considered</td>
<td>Not considered</td>
<td>Considered</td>
<td>Not considered</td>
<td>Considered (block section)</td>
<td>ILP, B&amp;C</td>
</tr>
<tr>
<td>This work</td>
<td>Considers</td>
<td>Not considers</td>
<td>Considers</td>
<td>Considers</td>
<td>Considers (minimum headway)</td>
<td>ILP, LR</td>
</tr>
</tbody>
</table>


constraints, such as track capacity and minimum headway time at joint points. Second, we design a two-layer time-space network for the considered problem. In this time-space network, several vertices for each repair track (parking track) are designed to represent the situation in which trains with different types can park at a repair track (parking track) at each time instant. This network is flexible and can be modified or extended for various practical operational requirements. Third, we propose a Lagrangian relaxation heuristic to solve the proposed model in which the feasible solution heuristic schedules trains one by one according to a ranked train order. To obtain the train order, we propose three ranking heuristics: a relaxed solution-based heuristic, a relatedness-based ranking heuristic, and a greedy heuristic.

The rest of this paper is organized as follows. In Section 2, we provide a detailed description of the considered problem. In Section 3, we introduce the two-layer time-space network representation and a corresponding multi-commodity network flow formulation of the problem. In Section 4, we describe our Lagrangian relaxation heuristic and feasible solution methods. Computational experiments are conducted to confirm the effectiveness of our network flow model and solution methods, and the results are reported in Section 5. Some conclusions are drawn in Section 6.
2 Problem Description

Figure 1 shows the topology of a typical EMU depot in China. It consists of a stabling yard, a cleaning area, a running shed, and a series of shunting track segments. The stabling yard has several parallel parking tracks (see, e.g., the depository track in Jacobsen and Pisinger 2011) on which trains can park to wait for daily maintenance or departure. The cleaning area includes one or more cleaning tracks along which equipment is installed to clean the trains. The running shed has several parallel repair tracks where central pits and overhead platforms are provided. Trains can park on repair tracks for daily maintenance in which some important parts of the EMU train, such as the cabs, carriages, pantographs, underframe equipment and wheels thereby can be checked and repaired carefully.

At China’s high-speed railways, as the repair, parking and station platforms are built to the same specifications. The car number in a feasible composition of EMU trains is limited which must have at least 8 but not more than 16, see, e.g., Zhong et al. (2019). In practice, EMUs can be roughly divided into two categories according to whether they contain 8 cars (i.e., short trains) or 16 cars (i.e., long trains), where a 16-car EMU train may consist of two 8-car EMU trains. In addition, train coupling and decoupling operations are usually not considered during the daily maintenance at CHSR. Each parking or repair track has two positions, as shown in Figure 1. Each position can be occupied by a maximum of one 8-car train at each time point. A 16-car train must occupy both position I and position II when on a parking or repair track. Each cleaning track can be occupied by a maximum of one train at any time. These parking tracks, repair tracks, and clean tracks are all called operation tracks because some operations may be implemented for trains on these tracks. In addition to these operation tracks, a series of shunting tracks are used to shunt trains among the depot’s various areas. Considering the safety operational constraints (see Section 2.2), the nodes along all of the tracks are also marked in the depot, as shown in Figure 1.

The general daily shunting process of a train in the depot proceeds as follows. After arriving at the depot, a train is first cleaned on a cleaning track. Next, maintenance is implemented on a repair track, and the train finally parks on a parking track to await its planned departure the next day. However, due to the limited number of cleaning tracks, a train may first park on a parking track before cleaning, or even undergo maintenance on the repair track before being cleaned on a cleaning track. Therefore, the train shunting problem in the depot studied here is defined as follows. Given a train depot and a set of trains with planned arrival and departure times, the problem aims to determine a shunting schedule for these trains that satisfies a series of maintenance requirements and safety operational requirements such as minimum headway constraints and track capacity constraints. The shunting schedule determines the assignment of operation tracks and the corresponding occupation timings to the trains, which also incorporates a detailed routing solution that represents a sequence of tracks (including both operation and shunting tracks) used and the time at which the trains enter and leave their assigned tracks.

We first state the assumptions of our study.

- There are two kinds of train lengths, one is for 8-cars EMUs and the other one is for 16-car EMUs.
• Each train has planned arrival and departure times at the depot.
• Train coupling and decoupling operations during the shunting process are not considered.
• The number of drivers for shunting services in the depot are sufficient and able to drive all types of trains, and trains thereby can move bidirectionally on tracks anytime without additional locomotives.
• All repair track segments are last-in-first-out tracks, that is, they can be approached from one side only.
• The depot is empty at both the beginning and ending times of the planning horizon.
• Limited by the considered depot topology, a train is only allowed to change its direction after being repaired if its maintenance activity in the depot is not cancelled, as this may benefit in reducing the number of shunting movements and the possibility of this train conflicting with other trains between the stabling yard and cleaning area. With this assumption, it is not allowed for a train to first go directly to the cleaning area, then be parked at the stabling yard for some time, and to finally be repaired at the running shed.

2.1 Input data

The planning horizon, denoted by \([0, T]\), is discretized, and the time units are expressed as integers (e.g., \(T = 1440\) if the planning horizon is 24 hours and each time unit is 1 minute). Table 2 summarizes the problem’s input parameters, where all time-related parameters are integer-valued.

2.1.1 Depot data

Let \(N = \{n_1, n_2, \ldots, n_{|N|}\}\) be the set of nodes marked in the depot, and let \(N_0 \subseteq N\) be the set of joint nodes in which each joint node connects three or more tracks (see node 3 in Figure 1). Note that one physical joint node between two crossover tracks is represented by two separate joint nodes in \(N_0\). For example, nodes 19 and 22 in Figure 1, which differs from the presentation in Caprara et al. (2011), where the route “... → 56 → 58 → 59 → ...” in their Figure 1 may be found. However, this route is infeasible, as a train cannot switch its travel direction in such a small crossover area. We collect these separate joint nodes in node set \(N_1\) and \(N_1 \subseteq N_0\) and denote \(\phi(i) = j\) and \(\phi(j) = i\) if and only if nodes \(i\) and \(j\)’s physical positions in the depot are the same. According to our representation, tracks \((28, 29), (29, 33), (31, 32),\) and \((32, 30)\) are considered, while tracks \((29, 30), (29, 31), (32, 28),\) and \((32, 33)\) do not exist in our study (see, Figure 1). Thus, some impossible paths (passing through tracks) in practical operations, such as paths \((31, 32) \rightarrow (32, 33)\) and \((28, 29) \rightarrow (29, 30)\), then do not exist in our solution, which may save the computational burden when exploring feasible solutions.

Let \(S = \{s_1, s_2, \ldots, s_{|S|}\}\) be the set of considered parking tracks in the stabling yard, \(R = \{r_1, r_2, \ldots, r_{|R|}\}\) be the set of considered repair tracks in the running shed, \(W = \{w_1, w_2, \ldots, w_{|W|}\}\) be the set of considered cleaning tracks in the cleaning area, and \(E = \{e_1, e_2, \ldots, e_{|E|}\}\) be the set of shunting tracks where each shunting track links two nodes in \(N\). The tracks mentioned above, including operation and shunting tracks, are all two-way tracks on which trains can move in both \(in\) and \(out\) directions.
2.1.2 Train data

Let $K_0$ and $K_1$ be the sets of short trains and long trains, respectively. Let $K$ be the set of all of the considered trains; i.e., $K = K_0 \cup K_1$. For each train $k \in K$, the input data include (i) the planned arrival time $t_a^k$ for train $k$ to the depot; (ii) the planned departure time $t_d^k$ for train $k$ from the depot; (iii) the required cleaning time $\alpha_k$; (iv) the required repair time $\beta_k$; (v) the minimum required time $\tau_p^k$ to shunt train $k$ to traverse track segment $p$, where $p \in S \cup R \cup W \cup E$; (vi) the operating cost $c_k$ incurred per time unit when the train is running on a track segment or undergoing operations on repair or cleaning tracks; and (vii) the operating cost $c'_k$ incurred per time unit when the train is waiting on a repair or cleaning track. Note that because operation tracks are very short, the time for drawing a train on an operation track $p \in S \cup R$ is mainly consumed in the ancillary work rather than in traversing the track. Hence, the time needed to shunt a train between any two points on the same operation track $p$ is set to be the same $\tau_p^k$. Using the parking track $s_0$ in Figure 1 as an example, the time needed to shunt a short train to traverse track $s_0$’s position I and the time needed to shunt a short train
to traverse the entire track \( s_0 \) (including both positions I and II) are the same \( r_{s_0}^k \).

Due to the limited maintenance capacity, a train’s maintenance work may not be finished before its planned departure time. With this consideration, we impose a penalty measurement on the maintenance schedule if a train’s actual departure time \( \hat{t}_{d_k}^d \) from the depot is later than the planned \( t_{d_k}^d \). For each train \( k \in K \), a penalty \( c''_k \zeta_k \) is imposed on train \( k \)’s shift \( \zeta_k \), which is defined as \( \max\{0, \hat{t}_{d_k}^d - t_{d_k}^d\} \). For each operational track, the value of time for repairing and cleaning trains is no less than that for parking trains, and shifting trains’ departure is worse than parking trains at operation tracks, we hereby set \( c'_k \leq c_k \leq c''_k \). In the worst-case scenario, train \( k \)’s maintenance requirement may be rejected by the depot; that is, train \( k \) is unscheduled, where a penalty of \( \pi_k \) is incurred.

2.2 Objective and constraints

The objective of the considered problem is to determine the shunting schedule and the routing solution for each train, such that the total cost is minimized. The route for train \( k \) begins and ends at a parking track; these two parking tracks may be different. A feasible solution must satisfy the following constraints.

- Daily maintenance requirements: Each train must be cleaned and repaired at the minimum required operation time during the planning horizon if this train’s maintenance activity is not cancelled.
- Headway constraints at nodes: For each node \( i \in N \), the arrivals of two trains at this node must be at least \( h_i \) time units apart, the departures of two trains from this node must be at least \( h_i \) time units apart, and one train’s arrival and another train’s departure from this node must be at least \( h_i \) time units apart.
- Track capacity constraints: Two or more trains may not occupy the same point on the track at the same time. For example, a train is not allowed to overtake another train on the track segment.

3 Time-Space Network Formulation

In this section, we formulate the considered problem as a minimum-cost multi-commodity network flow problem with several restrictions, where each commodity represents a train. The underlying network is an acyclic directed two-layer time-space network \( G = (V, A) \). In what follows, we first introduce the construction of our two-layer time-space network in Section 3.1. We then present the flow restrictions in Section 3.2. Finally, we propose the network flow formulation for the considered problem in Section 3.3.

3.1 Time-space network

In this section, we present the construction of our two-layer time-space network, including an inward layer and an outward layer (see Figure 2). A train traversing the arcs in the inward layer represents the situation in which this train is heading toward or entering the running shed, and a train traversing the arcs in the outward layer represents the situation in which this train is heading toward the station (see, for example, Figure 1).
The set of all possible time instants in the planning horizon is \( \{0, 1, \ldots, T\} \), which forms the “time” dimension of \( G \). The time dimension is the same in both the inward layer and the outward layer, whereas the “space” dimensions in the inward and outward layers are different. Specifically, in each “space” dimension we construct some components for parking, cleaning and repair tracks, as these components can represent different occupation states of these tracks. We also construct components for joint nodes, as the headway constraints (see Section 3.2) are particularly imposed on these joint nodes rather than on all the nodes marked in a depot layout. Details of the vertices and arcs in network \( G \) are introduced in what follows.

3.1.1 Vertices in inward layer network

For each parking track \( s \in S \), there are five components in the “space” dimension in the inward layer network, as shown in Figure 3. These components include: (i) \( \hat{\rho}_S(s) \), which represents a train’s arrival at parking track \( s \); (ii) \( \hat{\rho}_S(s) \), which represents a train’s occupation of position I of parking track \( s \), see, e.g., track (0,6) in Figure 1; (iii) \( \hat{\rho}_S(s) \), which represents a train’s occupation of position II of parking track \( s \), see, e.g., track (6,12) in Figure 1; (iv) \( \hat{\rho}_S'(s) \), which represents a train’s occupation of positions I and II of parking track \( s \), see, e.g., track (0,12) in Figure 1; and (v) \( \hat{\rho}_S(s) \), which represents a train’s departure from parking track \( s \).

For each cleaning track \( w \in W \), there are three components in the “space” dimension in the inward layer network. These components are as follows: (i) \( \hat{\rho}_W(w) \), which represents a train’s arrival at cleaning track \( w \); (ii) \( \rho_W(w) \), which represents a train’s movement on cleaning track \( w \); and (iii) \( \hat{\rho}_W(w) \) which represents a train’s occupation of cleaning track \( w \).

For each repair track \( r \in R \), there are seven components in the “space” dimension in the inward layer network, as shown in Figure 4. These components are as follows: (i) \( \hat{\rho}_R(r) \), which represents a train’s arrival at repair track \( r \); (ii) \( \rho_R(r) \), which represents a train’s arrival at position I of repair track \( r \); (iii) \( \hat{\rho}_R(r) \), which represents a train’s occupation of position I of repair track \( r \); (iv) \( \rho_R'(r) \), which represents a train’s arrival at position II of repair track \( r \); (v) \( \hat{\rho}_R'(r) \), which represents a train’s occupation of position II of repair track \( r \); (vi) \( \rho_R''(r) \), which represents a train’s arrival at positions I and II of repair track \( r \); and (vii) \( \hat{\rho}_R''(r) \), which represents a train’s occupation of positions I and II of repair track \( r \).
For each joint node \( i \in N_0 \), there is one component \( \rho(i) \) (see Figure 3) in the “space” dimension in the inward layer network. Mathematically, we let

\[
\Omega_{in} = \{ \tilde{\rho}_S(s), \bar{\rho}_S(s), \tilde{\rho}''_S(s), \hat{\rho}_S(s) \mid s \in S \} \cup \{ \tilde{\rho}_W(w), \rho_W(w), \bar{\rho}_W(w) \mid w \in W \}
\]

\[
\cup \{ \tilde{\rho}_R(r), \rho_R(r), \bar{\rho}_R(r), \tilde{\rho}''_R(r), \hat{\rho}_R(r) \mid r \in R \} \cup \{ \rho(i) \mid i \in N_0 \}
\]
denote the “space” dimension in the inward layer time-space network, and

\[
V_{in} = \{ (\omega, t) \mid \omega \in \Omega_{in}; t = 0, 1, \ldots, T \}
\]
denote the set of vertices in the inward layer time-space network.

### 3.1.2 Vertices in outward layer network

For each repair track \( r \in R \), there are four components in the “space” dimension in the outward layer network, as shown in Figure 4. These components are as follows: (i) \( \tilde{\rho}_R(r) \), which represents a train’s occupation of position I of repair track \( r \); (ii) \( \bar{\rho}'_R(r) \), which represents a train’s occupation of position II of repair track \( r \); (iii) \( \bar{\rho}''_R(r) \), which represents a train’s occupation of positions I and II of repair track \( r \); and (iv) \( \hat{\rho}_R(r) \), which represents a train’s departure from repair track \( r \).

For each parking track \( s \in S \), there are also five components in the “space” dimension in the outward layer network, as shown in Figure 5. These components are as follows: (i) \( \tilde{\varrho}_S(s) \), which represents a train’s arrival at parking track \( s \); (ii) \( \bar{\varrho}_S(s) \), which represents a train’s occupation of position I of parking track \( s \); (iii) \( \bar{\varrho}'_S(s) \), which represents a train’s occupation of position II of parking track \( s \); (iv) \( \bar{\varrho}''_S(s) \), which represents a train’s occupation of positions I and II of parking track \( s \); and (v) \( \hat{\varrho}_S(s) \), which represents a train’s departure from parking track \( s \).

For each cleaning track \( w \in W \), there are also three components in the “space” dimension in the outward layer network, as shown in Figure 5. These points are as follows: (i) \( \tilde{\varrho}_W(w) \), which represents a train’s arrival at cleaning track \( w \); (ii) \( \varrho_W(w) \), which represents a train’s movement on cleaning track \( w \); and (iii) \( \bar{\varrho}_W(w) \), which represents a train’s occupation of cleaning track \( w \).

For each joint node \( i \in N_0 \), there is also one component \( \varrho(i) \) (see Figure 5) in the “space” dimension in the outward layer network. Mathematically, we let

\[
\Omega_{out} = \{ \tilde{\varrho}_S(s), \bar{\varrho}_S(s), \tilde{\varrho}''_S(s), \hat{\varrho}_S(s) \mid s \in S \}
\]

\[
\cup \{ \tilde{\varrho}_W(w), \varrho_W(w), \bar{\varrho}_W(w) \mid w \in W \} \cup \{ \tilde{\varrho}_R(r), \bar{\varrho}'_R(r), \bar{\varrho}''_R(r), \hat{\varrho}_R(r) \mid r \in R \} \cup \{ \varrho(i) \mid i \in N_0 \}
\]
denote the “space” dimension in the outward layer time-space network, and let

\[
V_{out} = \{ (\omega, t) \mid \omega \in \Omega_{out}; t = 0, 1, \ldots, T \}
\]
denote the set of vertices in the outward layer time-space network.
Finally, the vertex set of the time-space network $G$ is

$$V = \{o,d\} \cup V_{in} \cup V_{out},$$

where vertex $o$ and vertex $d$ are the artificial source and the artificial sink, respectively, for the multi-commodity flow. In time-space network $G$, we can see that departure nodes for parking tracks exist in both inward and outward layers, while the departure nodes for repair tracks exist in the outward layer only, i.e., $\hat{\varrho}_R(r)$. Such a construction allows the practical last-in-first-out requirement for repair tracks to be implicitly met. As shown in Figure 4, after being repaired at parking tracks, trains will switch their direction from inward to outward via traversing switch arcs, and then wait at a specific position (position I, position II, or positions I and II) via traversing waiting arcs or be ready for departure via traversing departure arcs.

We can also see that there are no departure nodes for cleaning tracks in time-space network $G$. Recall that each cleaning track can be occupied by at most one EMU train (no matter if this train is a short or long one), while each repair or parking track has two positions which results in three occupation states that need to be considered, including position I or II is occupied by a short train, and both positions I and II are occupied by a long train. We construct departure nodes for parking and repairing tracks but not for cleaning tracks, as the movements on a track become complex if this track has two positions. For example, consider the following three situations on a parking track $s$: (i) a short train parks at position I ($\hat{\rho}_S(s)$) and then moves to position II ($\hat{\rho}''_S(s)$); (ii) a short train parks at position II ($\hat{\rho}'_S(s)$); and (iii) a long train parks at both positions I and II, it would occupy $\hat{\rho}''_S(s)$ rather than $\hat{\rho}'_S(s)$. Obviously, the short trains in situations (i) and (ii) can leave the parking track via the same node $\hat{\rho}'_S(s)$, while the long train in situation (iii) cannot leave the parking track using node $\hat{\rho}'_S(s)$. To make all trains leave the same track via the same node, we construct a departure node $\hat{\rho}_S(s)$ with which the constructed transfer arc originated from $\hat{\rho}_S(s)$ is unique. Practically, nodes $\hat{\rho}_W(w)$ and $\hat{\varrho}_W(w)$ can play the same role as departure nodes $\hat{\rho}_S(s)$, $\hat{\varrho}_S(s)$ and $\hat{\varrho}_R(r)$, as all of them can be used as the starting nodes for transfer arcs (see Appendix A).

### 3.1.3 Arcs

The arc set $A$ of the time-space network $G$ contains the following several types of arcs:

- **Starting arcs**: For the intermediate nodes between the station and depot, there exist some starting arcs which allow a train $k$ to start its operation in the depot at or after its planned arrival time $t_{ka}$.

- **Ending arcs**: For the intermediate nodes between the station and depot, there exist some ending arcs which allow a train $k$ to complete its operation in the depot at or after its planned departure time $t_{kd}$.

- **Drawing arcs**: For tracks in $S \cup R \cup W \cup E$, there exist some drawing arcs. A train traversing a drawing arc implies that this train is traversing the track that corresponds to the drawing arc.

- **Cleaning arcs**: For cleaning tracks, there exist some cleaning arcs, which allow a train to be cleaned with a minimum required cleaning time $\alpha_k$ when it dwells at a cleaning track and travels in the inward or outward direction.
• Repairing arcs: For repair tracks, there exist some repairing arcs, which allow a compatible train to be repaired with the minimum required time $\beta_k$ when it is dwelling on the repair track $r$.

• Waiting arcs: For tracks in $S \cup R \cup W$, there exist some waiting arcs, which allow a compatible train to park on the considered track.

• Departure arcs: For tracks in $S \cup R$, there exist some departure arcs, which represent the situation in which a train is about to leave the considered track.

• Transfer arcs: For tracks in $S \cup R \cup W$, there exist some transfer arcs, which allow trains to traverse the nodes between operation tracks and shunting tracks.

• Switch arcs: For repair tracks, there exist some switch arcs, which represent the situation in which a train has been repaired on the considered track and switches its travel from the inward direction to the outward direction. Note that these arcs change “layer” components from the inward layer to the outward layer.

• Dummy arc: There is a dummy arc $o \rightarrow d$ in network $G$. One can say that a train traversing this arc represents the situation in which this train’s daily maintenance requirement is rejected by the depot.

Figures 3, 4, and 5 illustrate different vertices and arcs described above. Note that these arcs have a common characteristic that each arc’s ending time instant is not earlier than its starting time instant. Hence, network $G$ is acyclic when all time-related parameters are positive. Moreover, there is a vector of cost coefficients $(\xi_{uv}^k, \ldots, \xi_{uv}^{K|})$ associated with each arc $u \rightarrow v \in A$, where cost coefficient $\xi_{uv}^k$ represents the cost for train $k$ to traverse arc $u \rightarrow v$. Each arc $u \rightarrow v \in A$ has a unit capacity per train. More details of the arcs described above and the corresponding cost coefficients are given in Appendix A. A path from vertex $o$ to vertex $d$ in our two-layer time-space network corresponds to a schedule of a train that determines the train’s sequence of operations and route (passing through track segments). Consider a feasible solution in which a short (respectively long) train arrives at the depot as shown in Figure 1, then traverses parking track $s_2$, track (14,18), track (18,21), track (21,24), parks at cleaning track $w_0$ for a cleaning operation, traverses track (26,28), (28,31), (31,35), dwells at position I (respectively positions I and II) of repair track $r_1$, tra-
verses tracks (35, 31), (31, 32), (32, 30), (30, 27), (27, 25), (25, 23), (23, 20), (20, 15) and then parks at position I (respectively positions I and II) of parking track s3 for planned departure. In the time-space network G, the corresponding order of nodes along the “space” dimension visited by a short train is \{o, \rho(2), \hat{\rho}_S(s_2), \check{\rho}_S(s_2), \rho(14), \rho(18), \rho(21), \rho(24), \hat{\rho}_W(w_0), \check{\rho}_W(w_0), \rho(26), \rho(28), \rho(31), \rho(35), \hat{\rho}_R(r_1), \check{\rho}_R(r_1), \rho(3), d\}, while the order of nodes along the “space” dimension visited by a long train is \{o, \rho(2), \hat{\rho}_S(s_2), \check{\rho}_S(s_2), \rho(14), \rho(18), \rho(21), \rho(24), \hat{\rho}_W(w_0), \check{\rho}_W(w_0), \rho(26), \rho(28), \rho(31), \rho(35), \hat{\rho}_R(r_1), \check{\rho}_R(r_1), \hat{\rho}_R(r_1), \check{\rho}_R(r_1), \rho(3), d\}.

### 3.2 Constraints

The schedule is feasible for train k if and only if all cost coefficients of train k along this path are finite. Given the constructed two-layer time-space network, our purpose is to determine feasible paths for all trains such...
that the total cost is minimized. However, these paths may not satisfy the constraints discussed in Section 2.2. Hence, in addition to the standard network flow constraints such as flow balance constraints and supply/demand constraints, our multi-commodity flow model also includes the following requirements and constraints:

- **Cleaning operation requirements:** For each $k \in K$, the train must be cleaned once during daily maintenance if it is not cancelled; we then impose the constraint that train $k$’s total flow along the cleaning arcs in the arc subset

  \[ B_W = A \cap \{o \to d\} \cup \{ (\tilde{\rho}_W(w), t) \to (\tilde{\rho}_W(w), t'), (\tilde{\rho}_W(w), t) \to (\tilde{\rho}_W(w), t') \mid w \in W; \ t, t' = 0, 1, \ldots, T\} \]

  must be one.

- **Repair operation requirements:** For each $k \in K$, the train must be repaired once during daily maintenance if it is not cancelled. Therefore, for each $k \in K_0$, we impose the constraint that train $k$’s total flow along the repairing arcs in the arc subset

  \[ B^k_R = A \cap \{o \to d\} \cup \{ (\rho^R(r), t) \to (\tilde{\rho}^R(r), t'), (\rho^R(r), t) \to (\tilde{\rho}^R(r), t') \mid r \in R; \ t, t' = 0, 1, \ldots, T\} \]

  must be one. For each $k \in K_1$, we impose the constraint that train $k$’s total flow along the repairing arcs in the arc subset

  \[ B^k_R = A \cap \{o \to d\} \cup \{ (\rho''_R(r), t) \to (\tilde{\rho}''_R(r), t') \mid r \in R; \ t, t' = 0, 1, \ldots, T\} \]

  must be one.

- **Headway constraints:** For each joint node $i \in N_0$, the time difference between two trains that traverse node $i$ must be at least $h_i$ units apart (see Section 2.2). For each $i \in N_0$ and each $t_1 = 0, 1, \ldots, T - h_i + 1$ we allow no more than one train to arrive at or depart from node $i$ during $[t_1, t_1 + h_i - 1]$. Denote

  \[ \hat{C}^{i1}_{it_1} = A \cap \{(\rho(i), t) \to (\tilde{\rho}_S(s), t), (\rho(i), t) \to (\tilde{\rho}_S(s), t) \mid s \in S; \ t = 0, 1, \ldots, T; \ t_1 \leq t \leq t_1 + h_i - 1\} \]

  \[ \cup \{ (\rho(i), t) \to (\tilde{\rho}_W(w), t), (\rho(i), t) \to (\tilde{\rho}_W(w), t) \mid w \in W; \ t = 0, 1, \ldots, T; \ t_1 \leq t \leq t_1 + h_i - 1\} \]

  \[ \cup \{ (\rho(i), t) \to (\tilde{\rho}_R(r), t), (\rho(i), t) \to (\tilde{\rho}_R(r), t) \to r \in R; \ t = 0, 1, \ldots, T; \ t_1 \leq t \leq t_1 + h_i - 1\} \]

  \[ \cup \{ (\rho(i), t) \to (\rho(j), t'), (\rho(i), t) \to (\rho(j), t') \mid j \in N \setminus \{i\}; \ t, t' = 0, 1, \ldots, T; \ t_1 \leq t \leq t_1 + h_i - 1\} \]

  Hence, for each node $i \in N_0 \setminus N_1$ and time instant $t_1 = 0, 1, \ldots, T - h_i + 1$, we impose the constraint that the total train flow along the arcs in the arc subset $\hat{C}^{i1}_{it_1}$ is at most one.

  For each node $i \in N_1$ and time instant $t_1 = 0, 1, \ldots, T - h_i + 1$, we impose the constraint that the total train flow along the arcs in the arc subset $\hat{C}^{i1}_{it_1} \cup \hat{C}^{j1}_{it_1}$ is at most one, where $j = \phi(i)$; that is, nodes $i$ and $j$ correspond to the same physical joint node between two crossover tracks in the depot (see, for example, nodes 29 and 32 in Figure 1). For convenience, denote

  \[ C^{i1}_{it_1} = \begin{cases} \hat{C}^{i1}_{it_1}, & \text{if } i \in N_0 \setminus N_1, t_1 = 0, 1, \ldots, T - h_i + 1; \\ \hat{C}^{i1}_{it_1} \cup \hat{C}^{j1}_{it_1}, & \text{if } i, j \in N_1 \text{ and } j = \phi(i), t_1 = 0, 1, \ldots, T - h_i + 1. \end{cases} \]
- Track capacity constraints: For each track segment in the depot, two or more trains cannot park in the same position of a track segment at any time. Hence, for each parking track $s \in S$ and time instant $t_1 = 0, 1, \ldots, T$, considering position I of parking track $s$, we impose the constraint that the total train flow along the arcs in the arc subset

$$C^2_{st_1} = A \cap \left[\{(\tilde{p}_S(s), t_1 - 1) \rightarrow (\tilde{p}_S(s), t_1), \ (\tilde{p}_S''(s), t_1 - 1) \rightarrow (\tilde{p}_S''(s), t_1), \right.$$ 

$$\left. (\tilde{g}_S(s), t_1 - 1) \rightarrow (\tilde{g}_S(s), t_1), \ (\tilde{g}_S''(s), t_1 - 1) \rightarrow (\tilde{g}_S''(s), t_1)\} \right.$$ 

$$\cup \left\{(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t'), \ (\tilde{p}_S''(s), t) \rightarrow (\tilde{p}_S''(s), t'), \right.$$ 

$$\left. (\tilde{g}_S(s), t) \rightarrow (\tilde{g}_S(s), t'), \ (\tilde{g}_S''(s), t) \rightarrow (\tilde{g}_S''(s), t') \ | \ t, t' = 0, 1, \ldots, T; t \leq t_1 < t'\right\}$$

is at most one. Considering position II of parking track $s$, we impose the constraint that the total train flow along the arcs in the arc subset

$$C^3_{st_1} = A \cap \left[\{(\tilde{p}_S(s), t_1 - 1) \rightarrow (\tilde{p}_S(s), t_1), \ (\tilde{p}_S''(s), t_1 - 1) \rightarrow (\tilde{p}_S''(s), t_1), \right.$$ 

$$\left. (\tilde{g}_S(s), t_1 - 1) \rightarrow (\tilde{g}_S(s), t_1), \ (\tilde{g}_S''(s), t_1 - 1) \rightarrow (\tilde{g}_S''(s), t_1)\} \right.$$ 

$$\cup \left\{(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t'), \ (\tilde{p}_S''(s), t) \rightarrow (\tilde{p}_S''(s), t'), \right.$$ 

$$\left. (\tilde{g}_S(s), t) \rightarrow (\tilde{g}_S(s), t'), \ (\tilde{g}_S''(s), t) \rightarrow (\tilde{g}_S''(s), t') \ | \ t, t' = 0, 1, \ldots, T; t \leq t_1 < t'\right\}$$

is at most one.

For each cleaning track $w \in W$ and time instant $t_1 = 0, 1, \ldots, T$, we impose the constraint that the total train flow along the arcs in the arc subset

$$C^4_{wt_1} = A \cap \left[\{(\tilde{p}_W(w), t_1 - 1) \rightarrow (\tilde{p}_W(w), t_1), \ (\tilde{p}_W(w), t_1 - 1) \rightarrow (\tilde{p}_W(w), t_1)\} \right.$$ 

$$\cup \left\{(\tilde{p}_W(w), t) \rightarrow (\tilde{p}_W(w), t'), \ (\tilde{p}_W(w), t) \rightarrow (\tilde{p}_W(w), t') \ | \ t, t' = 0, 1, \ldots, T; t \leq t_1 < t'\right\}$$

is at most one.

For each repair track $r \in R$ and time instant $t_1 = 0, 1, \ldots, T$, considering $r$’s position I, we impose the constraint that the total train flow along the arcs in the arc subset

$$C^5_{rt_1} = A \cap \left[\{(\tilde{p}_R(r), t_1 - 1) \rightarrow (\tilde{p}_R(r), t_1), \ (\tilde{p}_R''(r), t_1 - 1) \rightarrow (\tilde{p}_R''(r), t_1)\} \right.$$ 

$$\cup \left\{(\tilde{p}_R(r), t) \rightarrow (\tilde{p}_R(r), t'), \ (\tilde{p}_R(r), t) \rightarrow (\tilde{p}_R(r), t'), \right.$$ 

$$\left. (\tilde{p}_R(r), t) \rightarrow (\tilde{p}_R(r), t'), \ (\tilde{p}_R(r), t) \rightarrow (\tilde{p}_R(r), t') \ | \ t, t' = 0, 1, \ldots, T; t \leq t_1 < t'\right\}$$

is at most one. Considering $r$’s position II, we impose the constraint that the total train flow along the arcs
in the arc subset

\[ C_{rt}^6 = A \cap \left[ \{(\bar{\rho}'_R(r),t_1 - 1) \rightarrow (\bar{\rho}'_R(r),t_1), (\bar{\rho}''_R(r),t_1 - 1) \rightarrow (\bar{\rho}''_R(r),t_1)\} \right. \]

\[ \cup \left\{(\bar{\rho}'_R(r),t) \rightarrow (\bar{\rho}'_R(r),t'), (\bar{\rho}_R(r),t) \rightarrow (\bar{\rho}'_R(r),t'), \right. \]

\[ (\bar{\rho}'_R(r),t) \rightarrow (\bar{\rho}'_R(r),t'), (\bar{\rho}_R(r),t) \rightarrow (\bar{\rho}'_R(r),t'), \]

\[ \left. (\bar{\rho}''_R(r),t) \rightarrow (\bar{\rho}''_R(r),t'), (\bar{\rho}_R(r),t) \rightarrow (\bar{\rho}''_R(r),t') \mid t, t' = 0,1,\ldots,T; t \leq t_1 < t' \right\} \]

is at most one.

For each shunting track \((i,j) \in E\) and time instant \(t_1 = 0,1,\ldots,T\), we impose the constraint that the total train flow along the arcs in the arc subset

\[ C_{ijt}^7 = A \cap \{(\rho(i),t) \rightarrow (\rho(j),t'), (\theta(j),t) \rightarrow (\rho(i),t'') \mid t, t', t'' = 0,1,\ldots,T; t \leq t_1 < t' \}

is at most one.

Denote

\[ C = \{C_{it}^1 \mid i \in N_0; t = 0,1,\ldots,T - h_i + 1\} \cup \{C_{st}^2, C_{st}^3 \mid s \in S; t = 0,1,\ldots,T\}
\]

\[ \cup \{C_{wt}^4 \mid w \in W; t = 0,1,\ldots,T\} \cup \{C_{rt}^5, C_{rt}^6 \mid r \in R; t = 0,1,\ldots,T\} \cup \{C_{ijt}^7 \mid (i,j) \in E; t = 0,1,\ldots,T\}. \]

Then, for any arc set \(C \in C\), the total flow along the arcs in \(C\) cannot exceed one.

### 3.3 Integer programming formulation

For each train \(k \in K\) and \(u \rightarrow v \in A\), let \(x_{uv}^k = 1\) if arc \(u \rightarrow v\) is traversed by train \(k\), and \(x_{uv}^k = 0\) otherwise. If each train is taken as a commodity, then the considered problem can be formulated as a minimum-cost multi-commodity network flow formulation represented by the following integer program.

**P:** Minimize \(\sum_{k \in K} \sum_{u \rightarrow v \in A} x_{uv}^k \xi_{uv} \gamma_{uv}^k\) \hspace{1cm} (1)

subject to \(\sum_{(v:o \rightarrow u \in A)} x_{uv}^k = 1, \quad \text{for all } k \in K\) \hspace{1cm} (2)

\(\sum_{(u:d \rightarrow v \in A)} x_{uv}^k = 1, \quad \text{for all } k \in K\) \hspace{1cm} (3)

\(\sum_{(u:v \rightarrow w \in A)} x_{uw}^k = \sum_{(u:v \rightarrow w \in A)} x_{vw}^k \quad \text{for all } k \in K, v \in V \setminus \{o,d\}\) \hspace{1cm} (4)

\(\sum_{u \rightarrow v \in B_w} x_{uv}^k = 1, \quad \text{for all } k \in K\) \hspace{1cm} (5)

\(\sum_{k \in K} \sum_{u \rightarrow v \in C} x_{uv}^k \leq 1, \quad \text{for all } C \in C\) \hspace{1cm} (6)

\(x_{uv}^k \in \{0,1\}, \quad \text{for all } k \in K, u \rightarrow v \in A\) \hspace{1cm} (7)

The objective function (1) minimizes the total cost of the system for the schedule duration; that is, the total cost of operating services at the cleaning tracks and repair tracks, waiting for operations, and penalization on the shift of trains. Constraints (2) are the supply constraints that require the outflow of each train at vertex \(o\) to be 1. Constraints (3) are the demand constraints that require the inflow of each train at vertex \(d\) to be
1. Constraints (4) are the flow balance constraints for trains. Constraints (5) cover the cleaning operation requirement constraints introduced in Section 3.2. Constraints (6) cover the departure headway constraints, arrival constraints, and track capacity constraints introduced in Section 3.2. Constraints (7) are the binary constraints of the decision variables.

The following Proposition 1 implies that the repair operation requirements presented in Section 3.2 are implicitly satisfied in our network flow model, which benefits from the construction of the two-layer time-space network. Thus, there is no need to construct constraints related to repair operation requirements in model \( P \).

**Proposition 1** Repair operation requirements are always satisfied if problem \( P \) has a solution.

**Proof.** See Appendix B.

### 4 Lagrangian Relaxation Heuristic

In this section, we present the Lagrangian relaxation heuristic to solve the proposed model \( P \). Lagrangian relaxation has been widely and successfully used to solve transportation problems, see, e.g., Brännlund et al. (1998), Caprara et al. (2002; 2006), Cacchiani et al. (2012), and Jiang et al. (2017) for the train timetabling problem, and Dauzère-Pérès et al. (2015) for the integrated rolling stock units and train driver planning problem, as well as Mahmoudi and Zhou (2016) and Mahmoudi et al. (2021) for the vehicle routing problem.

#### 4.1 Lagrangian relaxation

We use the Lagrangian relaxation technique to relax constraints (5) and (6) of problem \( P \) and bring them into the objective function with associated Lagrangian multipliers \( \lambda_k \) (real number, \( k \in K \)) and \( \mu_C \geq 0 \) (\( C \in C \)). Let \( \lambda \) and \( \mu \) denote the vector of \( \lambda_k \) values and the vector of \( \mu_C \) values, respectively. The Lagrangian relaxed problem associated with the original optimization problem \( P \) can then be formulated as

\[
\tilde{\mathbf{P}}(\lambda, \mu) : \quad \text{Minimize} \quad \sum_{k \in K} \sum_{u \rightarrow v \in A} \xi_{uv}^k x_{uv}^k + \sum_{k \in K} \lambda_k (\sum_{u \rightarrow v \in B_w} x_{uv}^k - 1) \\
+ \sum_{C \in C} \mu_C (\sum_{k \in K} \sum_{u \rightarrow v \in C} x_{uv}^k - 1)
\]

subject to

- \[ \sum_{(v:o \rightarrow v \in A)} x_{ov}^k = 1, \quad \text{for all } k \in K \]
- \[ \sum_{(u:d \rightarrow v \in A)} x_{ud}^k = 1, \quad \text{for all } k \in K \]
- \[ \sum_{(u:v \rightarrow v \in A)} x_{uv}^k = \sum_{(v:v \rightarrow u \in A)} x_{vu}^k \quad \text{for all } k \in K, v \in V \setminus \{o, d\} \]
- \[ x_{uv}^k \in \{0, 1\}, \quad \text{for all } k \in K, u \rightarrow v \in A \]
After removing the constant \(-\sum_{k \in K} \lambda_k - \sum_{C \in C} \mu_C\) from the objective of problem \(\tilde{P}(\lambda, \mu)\), the reduced problem can be decomposed into \(|K|\) independent subproblems. For each \(k \in K\), the subproblem is

\[
\tilde{P}_k(\lambda, \mu) : \text{Minimize } \sum_{u \rightarrow v \in A} \xi_{uv}^k x_{uv}^k + \sum_{\{u \rightarrow v\} \in B_k} \lambda_k x_{uv}^k + \sum_{C \in C} \mu_C \sum_{u \rightarrow v \in C} x_{uv}^k
\]

subject to

\[
\sum_{\{v: o \rightarrow v\} \in A} x_{ov}^k = 1, \\
\sum_{\{u: u \rightarrow d\} \in A} x_{ud}^k = 1, \\
\sum_{\{u: u \rightarrow v\} \in A} x_{uv}^k = \sum_{\{v: v \rightarrow u\} \in A} x_{vu}^k \quad \text{for all } v \in V \setminus \{o, d\} \\
x_{uv}^k \in \{0, 1\}, \quad \text{for all } u \rightarrow v \in A
\]

Each subproblem \(\tilde{P}_k(\lambda, \mu)\) is a standard shortest path problem with arc length \(\delta_{uv}^k = \xi_{uv}^k + \sum_{\{u \rightarrow v: v \in B_k\}} \lambda_k + \sum_{\{C \in C: u \rightarrow v\} \in C} \mu_C\). This shortest path problem in the acyclic network \(G\) can be solved efficiently with a standard dynamic programming algorithm. Furthermore, these independent subproblems \(\tilde{P}_k(\lambda, \mu)\) can be solved in parallel when solved with a multicore computer processor. Given any nonnegative vectors \(\lambda\) and \(\mu\), we can obtain a lower bound on the optimal objective value of problem \(P\) by solving problem \(\tilde{P}(\lambda, \mu)\). Therefore, we can find a tight lower bound by solving the following problem

\[
\max_{\lambda, \mu} \tilde{P}(\lambda, \mu),
\]

which is referred to as the Lagrangian multiplier problem associated with the original optimization problem \(P\), see, e.g., Ahuja et al. (1993). In this work, we use a modified subgradient optimization technique to search for the near-optimal vector \(\lambda\) and vector \(\mu\); see Section 4.3 for more details.

### 4.2 Upper bound heuristic

In this section, we present a constructive heuristic for obtaining a feasible solution of the considered problem. In what follows, we first present the framework of our upper bound heuristic in Section 4.2.1 and then present three train order ranking heuristics in Section 4.2.2.

#### 4.2.1 Framework of the upper bound heuristic

Given a ranked train set \(\tilde{K} = (\tilde{k}_1, \tilde{k}_2, \ldots, \tilde{k}_{|K|})\), our upper bound heuristic derives a feasible solution of problem \(P\) as follows. We schedule these ranked trains \(k := \tilde{k}_1, \tilde{k}_2, \ldots, \tilde{k}_{|K|}\) one by one according to their ranks via the following six steps:

(i) similar to that in Caprara et al. (2002) and Xu et al. (2018), we first set the temporary cost \(\hat{\xi}_{uv}^k\) for train \(k\) to \(\delta_{uv}^k = \xi_{uv}^k + \sum_{\{u \rightarrow v: u \neq v\} \in B_k} \lambda_k + \sum_{\{C \in C: u \rightarrow v\} \in C} \mu_C\), and then revised \(\hat{\xi}_{uv}^k\) according to the paths of trains with higher ranks by the following equation.

\[
\hat{\xi}_{uv}^k = \begin{cases} 
+\infty, & \text{if } u \rightarrow v \text{ is incompatible with the paths of trains with higher ranks;} \\
\delta_{uv}^k, & \text{otherwise.}
\end{cases}
\]
(ii) fix the maintenance process as “first cleaning then repair” by setting \( \tilde{\xi}^k_{(\tilde{\varrho}W(w),t),(\tilde{\varrho}W(w),t')} = +\infty \) and 
\( \tilde{\xi}^k_{(\tilde{\varrho}W(w),t),(\varrho W(w),t')} = +\infty \) for each cleaning arc \( (\tilde{\varrho}W(w),t) \rightarrow (\tilde{\varrho}W(w),t') \) and each drawing arc \( (\varrho W(w),t) \rightarrow (\varrho W(w),t') \) in the inward layer network;

(iii) use a standard dynamic programming algorithm to find a shortest path for train \( k \), denoted by \( P^{(1)}_k \), from vertex \( o \) to vertex \( d \) in network \( G \), where the schedules of the trains with higher ranks than train \( k \) are kept unchanged;

(iv) reset \( \tilde{\xi}^k_{(\tilde{\varrho}W(w),t),(\varrho W(w),t')} \) and \( \tilde{\xi}^k_{(\varrho W(w),t),(\varrho W(w),t')} \) to their original values presented in Section 3.1.3, and set 
\( \tilde{\xi}^k_{(\tilde{\varrho}W(w),t),(\varrho W(w),t')} = +\infty \) and \( \tilde{\xi}^k_{(\varrho W(w),t),(\varrho W(w),t')} = +\infty \) for each cleaning arc \( (\tilde{\varrho}W(w),t) \rightarrow (\tilde{\varrho}W(w),t') \) and each drawing arc \( (\varrho W(w),t) \rightarrow (\varrho W(w),t') \) in the outward layer network to fix the maintenance process as “first repair then cleaning;”

(v) use a standard dynamic programming algorithm to find a shortest path for train \( k \), denoted by \( P^{(2)}_k \), from vertex \( o \) to vertex \( d \) in network \( G \), where the schedules of the trains with higher ranks than \( k \) are kept unchanged;

(vi) assign the shorter path in \( \{P^{(1)}_k, P^{(2)}_k\} \) to train \( k \).

Note that in steps (iii) and (v), we need to ensure that the obtained path does not violate any incompatibility constraint of problem \( P \). Because we use dynamic programming twice for each train \( k \), we then call the schedule generating method two-phase dynamic programming. See Algorithm 1 for a summary of this basic upper bound heuristic.

Algorithm 1 Upper Bound Heuristic

1: Input: a ranked train set with the order of \( (\tilde{k}_1, \tilde{k}_2, \ldots, \tilde{k}_{|K|}) \)
2: for \( k := \tilde{k}_1, \tilde{k}_2, \ldots, \tilde{k}_{|K|} \) do
3: apply the two-phase dynamic programming to construct a schedule for \( k \), where the schedules of the trains with higher ranks than \( k \) are kept unchanged
4: end for
5: Output: a feasible train schedule

4.2.2 Train order ranking methods

Because different priority sequences of the trains may result in train schedules with different qualities obtained by the above upper bound heuristic, we now present the following three methods to rank the priorities of trains.

Greedy heuristic: The greedy heuristic, GH for short, determines the priority sequence of trains following the first come first serve rule; that is, a train with an earlier arrival time at the depot is given higher priority in scheduling. As the train order determined by GH is fixed, we run the upper bound heuristic that incorporates GH only once to save computational time, i.e., the number of iterations in the subgradient optimization procedure (see Section 4.3) is one.

Relaxed solution-based heuristic: After obtaining the lower bound solution, the relaxed solution-based heuristic, RSH for short, ranks trains \( k \in K \) in nondecreasing order of the optimal objective values of \( \tilde{P}_k(\lambda, \mu) \).
**Relatedness-based ranking heuristic:** We now introduce a two-phase relatedness-based ranking heuristic, RRH for short, to re-rank the priority sequence of trains that are determined by RSH. Specifically, mimicking the way of Ropke and Pisinger (2006) and Shaw (1997), we define the relatedness measure $\Lambda(k, k')$ for each pair of trains $k$ and $k'$. The lower $\Lambda(k, k')$ is, the more related are these two trains, which means that there is greater interaction between these two trains. The relatedness measure defined in this paper consists of three terms: a path-length term, a time term and a type term. Given term weights $\alpha$, $\beta$ and $\gamma$, the relatedness for trains $i$ and $j$ is defined as

$$\Lambda(k, k') = \alpha(|l_k - l_{k'}|) + \beta(|t_{k}^{\text{d}} - t_{k'}^{\text{d}}| + |t_{k}^{d} - t_{k'}^{d}|) + \gamma(1 - \omega_{kk'})$$  \hspace{1cm} (8)

where $l_k$ denotes the length of train $k$’s path found in the lower bound solution. $\omega_{kk'} = 1$ if the trains $k$ and $k'$ are the same type; otherwise, $\omega_{kk'} = 0$. The proposed RRH randomly selects $|\varphi \cdot |K|/2|$ pairs of related trains, collocated in set $K$, with given parameters $\varphi \in (0, 1)$ and $\sigma \geq 1$ in the first phase. We select a pair of related trains via the following three steps:

(i) randomly select a train $k$ from train set $K'$ ($K'$ equals $K$ at the beginning) and delete this train from train set $K'$;

(ii) rank trains $k' \in K'$ in nondecreasing order of the values of relatedness $\Lambda(k, k')$;

(iii) randomly generate a number $y \in [0, 1)$, we select train pair of $k$ and the $\lfloor y^\sigma |K'|| \rfloor$-th train in the ranked train set $K'$ and insert this train pair, denoted by $(k, K'[\lfloor y^\sigma |K'|| \rfloor])$, into train pair set $K$.

In the second phase of the proposed RRH, we construct a new train order by swapping the positions of two related trains in a selected train pair in $\tilde{K}$. Algorithm 2 summarises this heuristic. The randomness existing in steps (i) and (iii) may result in different train orders that are determined by the RRH during different iterations of the Lagrangian relaxation heuristic, which may result in obtaining better feasible solutions.

**Algorithm 2** Relatedness-based Ranking Heuristic (RRH)

1: **Input:** a number $\varphi \in (0, 1)$, a determinism parameter $\sigma \geq 1$ and train order $\tilde{K} = (\tilde{k}_1, \tilde{k}_2, \ldots, \tilde{k}_{|K|})$ that is determined by RSH, an initialized train pair set $K = \emptyset$, and a train set $K' = K$

2: **Phase 1: Construct train pair set**

3:   **while** $|K| \leq |\varphi \cdot |K|/2|$ **do**

4:     step 1: randomly select a train $k$ from $K'$, set $K' \leftarrow K' \setminus \{k\}$

5:     step 2: rank $K'$ such that $k' < k'' \Rightarrow \Lambda(k, K'[k']) < \Lambda(k, K'[k''])$

6:     step 3: choose a random number $y$ from $[0, 1)$,

7:         update $K \leftarrow K \cup \{(k, K'[\lfloor y^\sigma |K'|| \rfloor])\}$, $K' \leftarrow K' \setminus \{K'[\lfloor y^\sigma |K'|| \rfloor]\}$

8: **end while**

9: **Phase 2: Construct new train order**

10:   **while** $|K| \neq 0$ **do**

11:     step 1: select one train pair $(k, k')$ from $K$

12:     step 2: update $K$ by swapping positions of trains $k$ and $k'$ in $K$

13:     step 3: update $K \leftarrow K \setminus \{(k, k')\}$

14: **Output:** re-ranked priority sequence of trains


20
4.3 Subgradient optimization procedure

In this section, we present a subgradient optimization procedure to search for near-optimal \( \lambda_k \) and \( \mu_C \) values. Note that the solution of problem \( \tilde{P}(\lambda, \mu) \) is a relaxed solution of problem \( P \) that may violate constraints (5) and (6) of problem \( P \). In each iteration of the subgradient optimization procedure, the value \( \sum_{u \rightarrow v \in B_u} x_{uv}^k - 1 \) for each \( k \in K \) and the value \( \sum_{k \in K} \sum_{u \rightarrow v \in C} x_{uv}^k - 1 \) for each \( C \in C \) form a subgradient vector \( \eta = \{ \eta_1, \ldots, \eta_{|K|}, \eta_{|K|+1}, \ldots, \eta_{|K|+|C|} \} \) of the relaxed solution. Let \( \eta^\ell (\lambda, \mu) \) denote the \( \eta \) (and \( \mu \)) vector in the \( \ell \)th iteration of the procedure, let \( \eta^\ell_m \) be the \( m \)th component of \( \eta^\ell \) for \( m = 1, 2, \ldots, |K| + |C| \), let \( \lambda^\ell_m \) be the \( m \)th component of Lagrangian multiplier vector \( \lambda^\ell \) for \( m = 1, 2, \ldots, |K| \), and let \( \mu^\ell_m \) be the \( m \)th component of Lagrangian multiplier vector \( \mu^\ell \) for \( m = 1, 2, \ldots, |C| \). In the initial iteration (i.e., \( \ell = 0 \)), the components in \( \eta \) are all initialized as 0, and the Lagrangian multipliers are all set to 0. In the other iterations (i.e., \( \ell > 0 \)), we update each multiplier according to the following formulas (see, e.g., Held and Karp 1971)

\[
\lambda^\ell_m \leftarrow \lambda^{\ell-1}_m + \theta \cdot \frac{UB - LB(\lambda, \mu)}{\|\eta^\ell\|^2} \cdot \eta^{\ell-1}_m \quad (m = 1, 2, \ldots, |K|)
\]

and

\[
\mu^\ell_m \leftarrow \max \left\{ \mu^{\ell-1}_m + \theta \cdot \frac{UB - LB(\lambda, \mu)}{\|\eta^\ell\|^2} \cdot \eta^{\ell-1}_m, 0 \right\} \quad (m = 1, 2, \ldots, |C|),
\]

where \( \theta > 0 \) is a prespecified step size parameter, \( UB \) is the best feasible solution of problem \( P \) found thus far, and \( LB(\lambda, \mu) \) is the optimal objective value of \( \tilde{P}(\lambda, \mu) \) corresponding to the current multipliers \( \lambda \) and \( \mu \).

To improve the convergence of the procedure and avoid the “zig-zag” behavior of the Lagrangian multipliers’ values, we further apply the modified subgradient technique proposed by Camerini et al. (1975). We use a modified subgradient vector \( \tilde{\eta} \) instead of \( \eta \) to update the Lagrangian multipliers \( \lambda_m \) and \( \mu_m \). In the \( \ell \)th (\( \ell > 0 \)) iteration, the modified subgradient vector \( \tilde{\eta}^\ell \) is updated by

\[
\tilde{\eta}^\ell \leftarrow \eta^\ell + b\tilde{\eta}^{\ell-1},
\]

where \( b \) is a scalar defined as

\[
b = \begin{cases} -a \cdot \frac{\tilde{\eta}^{\ell-1} \cdot \eta^\ell}{\|\tilde{\eta}^{\ell-1}\|^2}, & \text{if } \tilde{\eta}^{\ell-1} \cdot \eta^\ell < 0; \\ 0, & \text{otherwise}; \end{cases}
\]

and \( a \) is a prespecified value such that \( 0 \leq a \leq 2 \). The components in \( \tilde{\eta}^0 \) are all set to 0 in the initial iteration (i.e., \( \ell = 0 \)). Moreover, because the number of constraints (6) may be large in practice, we use a dynamic constraint-generation scheme to handle the relaxed constraints and to determine the corresponding multipliers. To implement this scheme, we initialize an empty constraint pool at first. In each iteration, the constraints violated by the relaxed solution are put into the constraint pool. The multipliers corresponding to these constraints in the pool are then updated with the method described above. After determining the multipliers, the constraints whose corresponding multipliers are updated to 0 are removed from the constraint pool.
In each iteration of the subgradient optimization procedure, we may implement the following five parts: (i) obtain a relaxed solution of problem $\mathbf{P}$ by solving $|K|$ shortest path problems $\tilde{\mathbf{P}}_k(\lambda, \mu)$; (ii) obtain a feasible solution of problem $\mathbf{P}$ using the upper bound heuristic presented in Section 4.2; (iii) identify constraints that the current relaxed solution has violated; (iv) update the subgradient vector and modified subgradient vector; and (v) update the Lagrangian multipliers. Because part (ii) is time-consuming, we may skip this part in some iterations to save computation time. This subgradient optimization procedure is terminated when one of the following two situations occurs: (i) the number of iterations reaches a prespecified limit; and (ii) the computational time reaches a prespecified limit. See Section 5 for more details regarding parameter settings.

## 5 Computational Study

In this section, we conduct a computational study to investigate the effectiveness of the network flow model and the performance of our Lagrangian heuristic. Our computational study has three parts. In the first part, we test our method on instances for two different depot networks, various train scales and timetable scenarios. In the second part, we test our solution method on instances to check whether the computational results are influenced by different ratios of short to long trains. In the third part, we examine our solution method on instances to analyze the performance of the relatedness-based ranking heuristic with various parameter settings. Our methods were implemented in C# using a personal computer with a 3.70 GHz 10-core processor (Intel Core i9-10900 Processor) and 32 GB RAM. In what follows, we first present the generation of the test instances in Section 5.1. We then report the result of computational study in Section 5.2.

### 5.1 Generation of test instances

In our computational study, we adopt a depot network at different scales. The first depot network, Network 1, is generated by capturing the structure characteristics of Hefei-Nan depot’s yard 2 in China’s high-speed railway, which has 60 nodes, 6 repair tracks, 1 cleaning track, 9 parking tracks, and 54 shunting tracks. The second depot network, Network 2, is a larger extension of Network 1 that has 163 nodes, 18 repair tracks, 3 cleaning tracks, 30 parking tracks, and 129 shunting tracks. Figures A1 and A2 in Appendix C, respectively, display Networks 1 and 2 in detail. We estimate the other train-related parameters such as the minimum arrival headway, cleaning time, repair time and unit operating costs by considering the characteristics of the trains operated by the Hefei-Nan depot.

We generate the data of each instance as follows, unless otherwise specified. For test instances underlying both Network 1 and Network 2, we set the length of the planning horizon to $T = 720$ minutes (e.g., from 18:00 one day to 06:00 the next), and the minimum headway between two trains that traverse the same node $i$ is set to 2 minutes, i.e., $h_i = 2$. In each instance, we generate the train data as follows. For each $k \in K$, we randomly select each train’s type from $\{0, 1\}$, where type 0 (represents a short train) is selected with a probability of 3/5, and type 1 (represents a long train) is selected with a probability of 2/5. For each short train $k \in K_0$
long train \( k \in K_1 \), the required cleaning time \( \alpha_k \) is set to 15 minutes (respectively 20 minutes), the required repair time \( \beta_k \) is set to 120 minutes (respectively 150 minutes). To capture the characteristic that trains undergo maintenance each night, we generate the trains’ arrival and departure times at the depot as follows. For each train \( k \), an arrival time \( t^a_k \) is randomly generated from a discrete uniform distribution between 0 and 360 (e.g., from 18:00 to 24:00) and its departure time \( t^d_k \) is randomly generated from a discrete uniform distribution between 600 and 720 (e.g., from 4:00 to 6:00). In addition, for each \( k \in K \), we set \( \tau^p_k = 2 \) minutes if \( p \in S \cup R \cup W \), and \( \tau^p_k = 1 \) minute if \( p \in E \). Note that many running activities over track segments could be lower than 1 min. However, our approach is general enough such that we can use a small discretized size if the time is lower than 1 but it may cost more computational time. Let \( c = 1.0 \) denote the unit operating cost for a short train \( k \in K_0 \) when it is waiting on a track. We then set \( c_k = (2.0)c \), \( c'_k = (1.0)c \), and \( c''_k = (2.4)c \) and for each train \( k \in K_0 \), we set \( c_k = (3.0)c \), \( c'_k = (1.5)c \), and \( c''_k = (3.6)c \) for each train \( k \in K_1 \). Moreover, we select a value between \([c'_k \times (\alpha_k + \beta_k), 2 \times c'_k \times (\alpha_k + \beta_k)]\) for \( \pi_k \), we hereby set \( \pi_k = (400.0)c \) and \( (750.0)c \) for each train in \( k \in K_0 \) and \( k \in K_1 \), respectively.

In our implementation of the Lagrangian heuristics, parameter \( a \) is set to 0.5 and parameter \( \theta \) is set to 2.0, and it will be reduced by 20% if the best lower bound identified shows no improvement for 20 consecutive iterations. For the solution of each test instance obtained by these methods, we determine their optimality gaps defined as

\[
\text{Gap} = \frac{UB - LB}{LB} \times 100\% ,
\]

where \( UB \) is the objective value obtained by the corresponding solution method and \( LB \) is the best lower bound value between the values obtained by Lagr. RSH and Lagr. RRH. The prespecified limit of computational time is set to 3 hours, and the prespecified limit of iterations is set to 2000 for each instance. For the first 300 iterations, we execute the upper bound heuristic and update the upper bound at each iteration. To save computation time, after 300 iterations, we execute the upper bound heuristic and update the upper bound with a probability of 20% at each iteration. In addition, in Lagr. RRH, we set \( \alpha = 0.2 \), \( \beta = 0.3 \), \( \gamma = 0.6 \), \( \varphi = 0.8 \), and \( \sigma = 2.0 \).

5.2 Computational results

5.2.1 Results with various train scales

For Network 1, we set the number of trains \( |K| = 4, 8, 12, 16, 20 \). Because Network 2 has a larger maintenance capacity, we set \( |K| = 8, 16, 24, 32, 40 \). We then have 10 combinations of depot network and number of trains. We generate five instances for each combination, wherein train timetables are generated independently. Thus, there are 50 instances with various train timetable scenarios. We use Lagr. RSH (Lagr. RRH) to denote the Lagrangian heuristic where RSH (RRH) is applied to determine the feasible solution of the problem P.

Table 3 summarizes the computational results of these test instances, with each row representing the results of the test instances. The “\(|K_0|\)” and “\(|K_1|\)” columns, respectively, report the mean number of short and long trains, the “\(|K_0^c|\)” column (\(|K_1^c|\)” reports the mean numbers of cancelled short trains (long trains), the “Gap”
column reports the mean value of the optimality gap, the “T(s)” column lists the mean value of time (in CPU second) to solve a test instance in each problem set. Table 3 shows that the optimality gaps obtained by all three heuristics tend to increase as $|K|$ increases and as the problem scale increases. Specifically, for these instances with less trains, the Lagrangian relaxation heuristic can output solutions with high quality, as there exist less incompatible constraints, see, e.g., instances S1, S2, S3, L1, L2, and L3. We also observe that limited by the depot capacity, a large number of trains leads to an increase in cancelled trains, which may in turn result in a larger optimality gap. This indicates either that the large-scale problem is not easy to solve or that the lower bound obtained by the Lagrangian relaxation heuristic is less tight for problems at a larger scale. Another natural result is that with an increase in problem scale, the required computational time for the Lagrangian heuristics increases significantly.

<table>
<thead>
<tr>
<th>Network 1</th>
<th>Network 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>inst</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$</td>
</tr>
<tr>
<td>S1</td>
<td>2.4</td>
</tr>
<tr>
<td>S2</td>
<td>5.4</td>
</tr>
<tr>
<td>S3</td>
<td>6.6</td>
</tr>
<tr>
<td>S4</td>
<td>9.4</td>
</tr>
<tr>
<td>S5</td>
<td>13.4</td>
</tr>
<tr>
<td>Avg. S</td>
<td>7.4</td>
</tr>
</tbody>
</table>

The average optimal gaps obtained by Lagr. RSH and Lagr. RRH are less than 5.70% (respectively 7.54%) for instances on Network 1 (respectively Network 2), while the average optimal gaps obtained by GH is 6.67% (respectively 8.97%) for instances on Network 1 (respectively Network 2). This computational result indicates that the solutions obtained by both Lagr. RSH and Lagr. RRH are better than that obtained by GH. This result illustrates the effectiveness of the proposed network flow model and the developed Lagrangian relaxation heuristic. We also observe that the solutions obtained by Lagr. RRH are slightly better than those obtained by Lagr. RSH in all test instances, implying that the RRH may find a better train schedule order than RSH. We can also see that GH can provide high-quality solutions for problems at a small scale, see, e.g., the first two instance groups in Networks 1 and 2. This may benefit from the tailored two-layer time-space network for the considered problem because it may make the problem easier to solve. In addition, the Lagrangian relaxation heuristic requires a much longer computation time than GH, especially for the large-scale problems, which shows that GH is highly efficient and indicates its utility in real-time applications.

The “T(s)” column of Table 3 shows that in all the test instances the CPU time consumed by Lagr. RSH...
and by Lagr. RRH is less than the prespecified 3 hours. For each test instance, the subgradient optimization procedure has 2000 iterations, wherein the upper bound heuristic is performed about 630 times. Figures 6(a) and 7(a) show the percentage of CPU time on the computation of each bound for the entire Lagr. RSH’s solution procedure for different instance sets, and Figures 6(b) and 7(b) show the percentage of average CPU time on the computation of each bound in one iteration. As expected, the time used in the feasible solution heuristic occupies more than half of the computational time, while the time used in the lower bound heuristic is the least amount. The CPU time percentage of the subgradient optimization tends to increase as the train scale increases, since there are more conflicts between train paths and more Lagrangian multipliers need to be updated in the instances with larger train scales. Moreover, there exist 175,680 (respectively 514,080) nodes and 287,280 (respectively 876,240) arcs in the time-space network that is constructed for each instance on Network 1 (respectively 2). Figure 6(c) shows the average numbers of arcs existing in relaxed shortest paths vary over a small range, while the average numbers of arcs existing in feasible shortest paths, save for S1, tends to increase as the number of trains increases as the situation becomes denser in Network 1. Figure 7(c) shows the average numbers of arcs existing in both feasible and relaxed shortest paths vary over a small range and follow a similar tendency for the instances on Network 2.

Figure 8 plots the convergence of a test instance in problem set L5, which shows the result obtained by Lagr. RSH (Lagr. RRH), the optimality gap improves from 35.16% to 12.75% (12.49%), the best upper and
lower values improve by 1.58% and 11.06% (1.71% and 10.60%), respectively, when the iteration number is increased from 1 to 2000. This demonstrates that the lower bound can become much tighter and the Lagrangian relaxation heuristic can generate solutions with much smaller performance gap if we increase the iteration number significantly. We can also observe that the upper bounds become stable around the 1100th iteration. We have selected a test instance from problem set S4 to investigate the schedule details, where 12 short and 8 long trains are taken into consideration. Appendix D displays the depot schedule obtained by Lagr. RSH in detail for the selected instance. Scheduling 20 trains on depot Network 1 is a difficult task, where 2 short and 3 long trains are not scheduled due to the small capacity of depot Network 1.

5.2.2 Results with various ratios of short to long trains

In this section, we consider six ratios of the number of short trains to that of long trains: $R_1 = 0 : 5$, $R_2 = 1 : 4$, $R_3 = 2 : 3$, $R_4 = 3 : 2$, $R_5 = 4 : 1$ and $R_6 = 5 : 0$. Considering that there are 9 and 30 parking tracks in Networks 1 and 2, the train scales for the instances with Networks 1 and 2 are, respectively, set to 12 and 32 which is slightly greater than the number of parking tracks. For each ratio, we generate five instances. Thus, there are 60 instances with various train timetable scenarios. Table 4 presents the computational results wherein the “$\bar{\zeta}$” column reports the value of the average shift of the not cancelled trains in each test instance set.

Table 4 indicates that GH has more probability to reject maintenance requirements of trains, see “$K_0$” and “$K_1$” columns, implying that the Lagrangian dual information for the feasible heuristic in Lagr. RSH helps improve the quality of the depot schedule. The “Gap” column shows that the optimality gap tends to decrease as the percentage of short trains increases. Since the same depot network can accommodate more short trains...
Table 4: Computational results with various ratios of short to long trains.

<table>
<thead>
<tr>
<th>Network 1</th>
<th>GH</th>
<th>Lagr. RSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-R1 0.0</td>
<td>12.0</td>
<td>7232.5</td>
</tr>
<tr>
<td>S-R2 2.0</td>
<td>10.0</td>
<td>6760.2</td>
</tr>
<tr>
<td>S-R3 4.6</td>
<td>7.4</td>
<td>5942.2</td>
</tr>
<tr>
<td>S-R4 8.0</td>
<td>4.0</td>
<td>4794.4</td>
</tr>
<tr>
<td>S-R5 10.8</td>
<td>1.2</td>
<td>4028.7</td>
</tr>
<tr>
<td>S-R6 12.0</td>
<td>0.0</td>
<td>3753.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network 2</th>
<th>GH</th>
<th>Lagr. RSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>L-R1 0.0</td>
<td>32.0</td>
<td>20507.4</td>
</tr>
<tr>
<td>L-R2 5.0</td>
<td>27.0</td>
<td>19086.4</td>
</tr>
<tr>
<td>L-R3 13.4</td>
<td>18.6</td>
<td>16581.3</td>
</tr>
<tr>
<td>L-R4 17.0</td>
<td>15.0</td>
<td>15480.6</td>
</tr>
<tr>
<td>L-R5 25.8</td>
<td>6.2</td>
<td>12859.6</td>
</tr>
<tr>
<td>L-R6 32.0</td>
<td>0.0</td>
<td>11073.4</td>
</tr>
</tbody>
</table>

as the short trains require less space in the depot than that required by the long trains. In addition, we see that there exist some train shifts in 9 out of the 12 instance sets solved by Lagr. RSH, but the values of the train shifts are small. Typically, a train shift may be more acceptable than a train cancellation. In practice, we may sacrifice the punctuality of the train schedule to reduce the number of cancelled trains.

5.2.3 Results with various parameters $\varphi$ and $\sigma$

Next, we conduct a computational study of sensitivity analysis to show how the computational results obtained by the relatedness-based ranking heuristic are affected by the values of parameter $\varphi$ and parameter $\sigma$. In this computational study, we generate two instances using the same method of generating problem sets S5 and L5. The first instance with underlying Network 1 considers 15 short trains and 5 long trains. The second instance with underlying Network 2 considers 22 short trains and 18 long trains. For each instance, we set $\varphi = 0.2, 0.4, 0.6, 0.8$ and $\sigma = 1, 2, 3, 4$. There are thus 16 combinations of $\varphi$ and $\sigma$, and 32 computational instances. The other parameter settings are the same as in the first experimental set.

Table 5: Computational results with different $\varphi$ and $\sigma$ (20 trains on Network 1).

<table>
<thead>
<tr>
<th>$\varphi$</th>
<th>$\sigma = 1$</th>
<th>$\sigma = 2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varphi$</td>
<td>$\sigma = 3$</td>
<td>$\sigma = 4$</td>
</tr>
</tbody>
</table>

27
Tables 5 and 6 summarize the computational results of the sensitivity analysis study, where optimality gaps \( \text{Gap}_1 \) and \( \text{Gap}_2 \) are, respectively, defined as

\[
\text{Gap}_1 = \frac{UB - LB_1}{LB_1} \times 100\%, \quad \text{Gap}_2 = \frac{UB - LB_2}{LB_2} \times 100%,
\]

where \( UB \) and \( LB_1 \) are, respectively, the objective value of corresponding solution and the lower bound value in the instance with the same setting of \( \phi \) and \( \sigma \), while \( LB_2 \) (see the value with over line in both Table 5 and Table 6) is the best lower bound value in these instances with different settings of \( \phi \) and \( \sigma \). We can see that in the results with underlying Network 1, the maximum optimality gap is 12.9%, while the minimum optimality gap is 9.5%. This indicates that the values of \( \phi \) and \( \sigma \) can affect the performance of Lagr. RRH.

### Table 6: Computational results with different \( \phi \) and \( \sigma \) (40 trains on Network 2).

| \( \phi \) | \( \sigma \) | \( LB_1 \) | \( UB \) | \( |K_0^c| \) | \( |K_1^c| \) | \( \bar{\zeta} \) | \( \text{Gap}_1 \) | \( \text{Gap}_2 \) | \( T(s) \) |
|---|---|---|---|---|---|---|---|---|---|
| 0.2 | 1 | 17312.2 | 19181.6 | 2 | 1 | 0.00 | 10.8% | 10.6% | 7842 |
| 0.4 | 1 | 17336.5 | 19253.1 | 2 | 1 | 0.00 | 11.1% | 11.1% | 7522 |
| 0.6 | 1 | 17266.5 | 19230.3 | 2 | 1 | 0.00 | 11.4% | 10.9% | 7499 |
| 0.8 | 1 | 17216.6 | 19257.3 | 2 | 1 | 0.00 | 11.9% | 11.1% | 7542 |

| \( \phi \) | \( \sigma \) | \( LB_1 \) | \( UB \) | \( |K_0^c| \) | \( |K_1^c| \) | \( \bar{\zeta} \) | \( \text{Gap}_1 \) | \( \text{Gap}_2 \) | \( T(s) \) |
|---|---|---|---|---|---|---|---|---|---|
| 0.2 | 1 | 17296.1 | 19261.0 | 2 | 1 | 0.00 | 11.4% | 11.1% | 7677 |
| 0.4 | 1 | 17302.7 | 19239.3 | 2 | 1 | 0.00 | 11.2% | 11.0% | 7539 |
| 0.6 | 1 | 17235.2 | 19244.5 | 2 | 1 | 0.00 | 11.7% | 11.0% | 7376 |
| 0.8 | 1 | 17248.6 | 19204.2 | 2 | 1 | 0.00 | 11.9% | 10.8% | 7637 |

We can also see that in the results with underlying Network 2, the maximum optimality gap is 12.0%, while the minimum optimality gap is 10.8%. That is, there exists a less difference among the optimality gaps in the results with underlying Network 2. Moreover, we can also see that the difference between \( \text{Gap}_1 \) and \( \text{Gap}_2 \) for these instances on Network 1 is larger than that on Network 2. A comparison of the results with the underlying Network 1 and the results with the underlying Network 2 indicates that the number of cancelled trains in the former is larger than that in the latter, even though the former has less trains than the latter. This means the situation with 20 trains on Network 1 is denser than that with 40 trains on Network 2. Combining the above observations, we can infer that Lagr. RRH’s performance may be highly influenced by the values of \( \phi \) and \( \sigma \) in a dense situation. In addition, the results in Table 5 allows us to roughly conclude that the optimality gaps obtained by Lagr. RRH tend to decrease as the value of \( \phi \) increases.

## 6 Conclusions

This study examined the model and solution method for the passenger train shunting and routing problem in a depot. We presented a minimum-cost multi-commodity network flow model for the considered problem on the basis of a tailored two-layer time-space network. We then developed Lagrangian relaxation heuristics and a
benchmark solution method to solve the problem. The computational results demonstrate the effectiveness of the proposed model and the solution methods.

The construction of a two-layer time-space network was an important part of this study. For example, we carefully designed several vertices for each repair track and each parking track to deal with a practical operation requirement that allows only one long train or two short trains to park on a repair or parking track at each time instant. Our network is flexible and can be easily modified to accommodate other practical requirements. For example, if wheel repair is required, we can design vertices and arcs corresponding to a wheel repair track following the method to design vertices and arcs corresponding to cleaning tracks, because wheel repair equipment is also installed along tracks rather than in a running shed. In addition, in this study, we considered only one kind of depot layout in China’s high-speed railways. An interesting extension of our work would be to apply our solution method to other depots with different layouts by modifying our time-space network according to the considered layout.

It is worth mentioning that some uncertainty factors exist; for example, the train’s arrival time at the depot may be delayed by malfunctioning infrastructure or rolling stock, and these disturbances may make the schedule less efficient or even infeasible. It would thus be worthwhile to study the optimization model and solution method for scheduling trains in the depot to allow an optimal response to such disturbances.

**Acknowledgement**

This study is supported by the National Natural Science Foundation of China (No. 72071059, 72188101) and China Postdoctoral Science Foundation (No. 2019M662144). We thank Mr. Wei Zhang from Hefei Railway Station for his help on data collection and problem definition.

**References**


Appendix A. Description of Arcs

- Starting arcs: For each node $i \in N$ and time instant $t = 0, 1, \ldots, T$, there is a starting arc $o \rightarrow (\rho(i), t)$ if this node is a joint node between the station and depot (i.e., nodes 2 and 3 in Figure 1). For each train $k \in K$, if $t \geq t^k_b$, then $\xi^k_{s_{\rho(i)}, t} = 0$; otherwise $\xi^k_{s_{\rho(i)}, t} = +\infty$.

- Ending arcs: For each node $i \in N$ and time instant $t = 0, 1, \ldots, T$, there is an ending arc $(\rho(i), t) \rightarrow d$ if this node is a joint node between the station and depot. As mentioned in Section 2.1.2, a shift cost is imposed if a train’s actual departure time is later than planned. Considering this, we impose the shift cost $\Gamma_k(t) = \max \{0, c^k_b(t - t^k_b)\}$ on an ending arc. For each train $k \in K$, $\xi^k_{(\rho(i), t), d} = \Gamma_k(t)$ if $t^k_b \leq t \leq T$; otherwise, $\xi^k_{(\rho(i), t), d} = +\infty$.

- Drawing arcs: For each parking track $s \in S$ and time instants $t, t' = 0, 1, \ldots, T$, there are three drawing arcs, including $(\bar{p}_s(s), t) \rightarrow (\bar{p}_s(s), t')$, $(\bar{p}_s(s), t) \rightarrow (\bar{p}'_s(s), t')$, and $(\bar{p}_s(s), t) \rightarrow (\bar{p}_{\rho}(s), t')$, in the inward layer network and three drawing arcs, including $(\bar{d}_s(s), t) \rightarrow (\bar{d}_s(s), t')$, $(\bar{d}_s(s), t) \rightarrow (\bar{d}'_s(s), t')$, and $(\bar{d}_s(s), t) \rightarrow (\bar{d}_{\rho}(s), t')$, in the outward layer network if there exists $k \in K_0$ such that $t \geq t^k_b$ and $t' = t + \tau^k_s \leq T$. For each train $k \in K$ and each drawing arc $u \rightarrow v$ mentioned above, if $k \in K_0$, $t \geq t^k_b$ and $t' = t + \tau^k_s \leq T$, then $\xi^k_{u, v} = c_k(t' - t)$ where $t$ and $t'$ are the time instants corresponding to vertices $u$ and $v$. There is one drawing arc $(\bar{p}_s(s), t) \rightarrow (\bar{p}'_s(s), t')$ in the inward layer network and one drawing arc $(\bar{d}_s(s), t) \rightarrow (\bar{d}'_s(s), t')$ in the outward layer network if there exists $k \in K_1$ such that $t \geq t^k_b$ and $t' = t + \tau^k_s \leq T$. For each train $k \in K$, if $k \in K_1$, $t \geq t^k_b$ and $t' = t + \tau^k_s \leq T$, then $\xi^k_{(\bar{p}_s(s), t), (\bar{p}'_s(s), t')} = \xi^k_{(\bar{d}_s(s), t), (\bar{d}'_s(s), t')} = c_k(t' - t)$; otherwise, $\xi^k_{(\bar{d}_s(s), t), (\bar{d}'_s(s), t')} = \xi^k_{(\bar{d}_s(s), t), (\bar{d}_s(s), t')} = +\infty$.

For each repair track $r \in R$ and time instants $t, t' = 0, 1, \ldots, T$, there are three drawing arcs $(\rho_R(r), t) \rightarrow (\rho'_R(r), t')$, $(\rho_R(r), t) \rightarrow (\rho''_R(r), t')$, and $(\rho_R(r), t) \rightarrow (\rho_R(r), t')$ if there exists $k \in K_0$ such that $t \geq t^k_b$ and $t' = t + \tau^k_r \leq T$. For each train $k \in K$, if $k \in K_0$, $t \geq t^k_b$ and $t' = t + \tau^k_r \leq T$, then $\xi^k_{(\rho_R(r), t), (\rho'_R(r), t')} = \xi^k_{(\rho_R(r), t), (\rho''_R(r), t')} = \xi^k_{(\rho_R(r), t), (\rho_R(r), t')} = c_k(t' - t)$; otherwise, $\xi^k_{(\rho_R(r), t), (\rho'_R(r), t')} = \xi^k_{(\rho_R(r), t), (\rho''_R(r), t')} = +\infty$.

For each cleaning track $w \in W$ and time instants $t, t' = 0, 1, \ldots, T$, there are two drawing arcs $(\rho_w(w), t) \rightarrow (\rho_w(w), t')$ and $(\rho_w(w), t) \rightarrow (\rho'_w(w), t')$ if there exists $k \in K$ such that $t \geq t^k_b$ and $t' = t + \tau^k_w \leq T$. For each train $k \in K$, if $k \in K$, $t \geq t^k_b$ and $t' = t + \tau^k_w \leq T$, then $\xi^k_{(\rho_w(w), t), (\rho_w(w), t')} = \xi^k_{(\rho_w(w), t), (\rho'_w(w), t')} = c_k(t' - t)$; otherwise, $\xi^k_{(\rho_w(w), t), (\rho'_w(w), t')} = \xi^k_{(\rho_w(w), t), (\rho_w(w), t')} = +\infty$.

For each shunting track $e = (i, j) \in E$ and time instants $t, t' = 0, 1, \ldots, T$, there are two drawing arcs $(\rho(i), t) \rightarrow (\rho(j), t')$ and $(\rho(j), t) \rightarrow (\rho(i), t')$ if there exists $k \in K$ such that $t \geq t^k_b$ and $t' = t + \tau^k_e \leq T$. For each train $k \in K$, if $k \in K$, $t \geq t^k_b$ and $t' = t + \tau^k_e \leq T$, then $\xi^k_{(\rho(i), t), (\rho(j), t')} = \xi^k_{(\rho(j), t), (\rho(i), t')} = c_k(t' - t)$; otherwise, $\xi^k_{(\rho(i), t), (\rho(j), t')} = \xi^k_{(\rho(j), t), (\rho(i), t')} = +\infty$.

A train traversing a drawing arc implies that this train is traversing the track that corresponds to the drawing.
• Cleaning arcs: For each $k \in K$, $w \in W$, and $t, t' = 0, 1, \ldots, T$, there are two cleaning arcs $(\tilde{p}_W(w), t) \rightarrow (\tilde{p}_W(w), t')$ and $(\tilde{q}_W(w), t) \rightarrow (\tilde{q}_W(w), t')$ if $t \geq t_k^a$ and $t' = t + \alpha_k \leq T$. For each train $k \in K$, if $t \geq t_k^a$ and $t' = t + \alpha_k \leq T$, then $\xi_{(\tilde{p}_W(w), t), (\tilde{p}_W(w), t')}^k = \xi_{(\tilde{q}_W(w), t), (\tilde{q}_W(w), t')}^k = c_k(t' - t)$; otherwise, $\xi_{(\tilde{p}_W(w), t), (\tilde{p}_W(w), t')}^k = \xi_{(\tilde{q}_W(w), t), (\tilde{q}_W(w), t')}^k = +\infty$. The arc $(\tilde{p}_W(w), t) \rightarrow (\tilde{p}_W(w), t')$ allows train $k$ to be cleaned with a minimum required cleaning time $\alpha_k$ when it travels at cleaning track $w$ and travels in the inward direction. The clean arc $(\tilde{q}_W(w), t) \rightarrow (\tilde{q}_W(w), t')$ allows train $k$ to be cleaned with a minimum required cleaning time $\alpha_k$ when it travels at cleaning track $w$ and travels in the outward direction.

• Repairing arcs: For each $k \in K_0$, $r \in R$ and $t, t' = 0, 1, \ldots, T$, there are two repairing arcs $(\rho_R(r), t) \rightarrow (\tilde{p}_R(r), t')$ and $(\rho_R(r), t) \rightarrow (\tilde{p}_R(r), t')$ if $t \geq t_k^a$ and $t' = t + \beta_k \leq T$. For each train $k \in K$, if $k \in K_0$, $t \geq t_k^a$ and $t' = t + \beta_k \leq T$, then $\xi_{(\rho_R(r), t), (\tilde{p}_R(r), t')}^k = \xi_{(\rho_R(r), t), (\tilde{p}_R(r), t')}^k = c_k(t' - t)$; otherwise, $\xi_{(\rho_R(r), t), (\tilde{p}_R(r), t')}^k = \xi_{(\rho_R(r), t), (\tilde{p}_R(r), t')}^k = +\infty$. The arc $(\rho_R(r), t) \rightarrow (\tilde{p}_R(r), t')$ (arc $(\rho_R(r), t) \rightarrow (\tilde{p}_R(r), t')$) allows short train $k$ to be repaired with the minimum required repairing time $\beta_k$ when it is dwell at repair track $r$’s position I (position II). For each $k \in K_1$, $r \in R$ and $t, t' = 0, 1, \ldots, T$, there is also a repairing arc $(\rho_R^1(r), t) \rightarrow (\tilde{p}_R^1(r), t')$ if $t \geq t_k^a$ and $t' = t + \beta_k \leq T$. For each train $k \in K$, if $k \in K_1$, $t \geq t_k^a$ and $t' = t + \beta_k \leq T$, then $\xi_{(\rho_R^1(r), t), (\tilde{p}_R^1(r), t')}^k = c_k(t' - t)$; otherwise, $\xi_{(\rho_R^1(r), t), (\tilde{p}_R^1(r), t')}^k = +\infty$. This arc allows long train $k$ to be repaired with the minimum required time $\beta_k$ when it travels along the entire repair track $r$.

• Waiting arcs: For each parking track $s \in S$ and time instant $t = 0, 1, \ldots, T - 1$, there are four waiting arcs, including $(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$, $(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$, $(\tilde{q}_S(s), t) \rightarrow (\tilde{q}_S(s), t + 1)$, and $(\tilde{q}_S(s), t) \rightarrow (\tilde{q}_S(s), t + 1)$ if there exists $k \in K_0$ such that $t \geq t_k^a$, and there are two waiting arcs $(\rho_S^1(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$, and $(\rho_S^1(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$ if there exists $k \in K_1$ such that $t \geq t_k^a$. For each $k \in K$, if $k \in K_0$, $t \geq t_k^a$, $\xi_{(\rho_S(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\rho_S(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\rho_S(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\rho_S(s), t), (\tilde{p}_S(s), t + 1)}^k = 0$; otherwise, $\xi_{(\tilde{p}_S(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\tilde{p}_S(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\tilde{p}_S(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\tilde{p}_S(s), t), (\tilde{p}_S(s), t + 1)}^k = +\infty$. The arc $(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$ (respectively arc $(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$) allows a short train to wait at parking track $s$’s position I (position II) when traveling in the inward direction. The arc $(\tilde{q}_S(s), t) \rightarrow (\tilde{q}_S(s), t + 1)$ (arc $(\tilde{q}_S(s), t) \rightarrow (\tilde{q}_S(s), t + 1)$) allows a short train to wait at parking track $s$’s position I (position II) when traveling in the outward direction. For each $k \in K$, if $k \in K_1$, $t \geq t_k^a$, $\xi_{(\rho_S^1(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\rho_S^1(s), t), (\tilde{p}_S(s), t + 1)}^k = 0$; otherwise, $\xi_{(\tilde{p}_S(s), t), (\tilde{p}_S(s), t + 1)}^k = \xi_{(\tilde{p}_S(s), t), (\tilde{p}_S(s), t + 1)}^k = +\infty$. The arc $(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$ (arc $(\tilde{p}_S(s), t) \rightarrow (\tilde{p}_S(s), t + 1)$) allows a long train to wait on the entire parking track $s$ when traveling in the inward (outward) direction.

For each cleaning track $w \in W$ and time instant $t = 0, 1, \ldots, T - 1$, there are two waiting arcs $(\tilde{p}_W(w), t) \rightarrow (\tilde{p}_W(w), t + 1)$ and $(\tilde{q}_W(w), t) \rightarrow (\tilde{q}_W(w), t + 1)$ if there exists train $k \in K$ such that $t \geq t_k^a$. For each $k \in K$, if $t \geq t_k^a$, then $\xi_{(\tilde{p}_W(w), t), (\tilde{p}_W(w), t + 1)}^k = \xi_{(\tilde{q}_W(w), t), (\tilde{q}_W(w), t + 1)}^k = c_k^r$; otherwise, $\xi_{(\tilde{p}_W(w), t), (\tilde{p}_W(w), t + 1)}^k = \xi_{(\tilde{q}_W(w), t), (\tilde{q}_W(w), t + 1)}^k = +\infty$. The arc $(\tilde{p}_W(w), t) \rightarrow (\tilde{p}_W(w), t + 1)$ and arc $(\tilde{q}_W(w), t) \rightarrow (\tilde{q}_W(w), t + 1)$ allow a train to wait at the cleaning track $w$ when traveling in the inward and outward directions, respectively.
For each repair track \( r \in R \) and time instant \( t = 0, 1, \ldots, T - 1 \), there are two waiting arcs \((\bar{g}_R(r), t) \rightarrow (\bar{g}_R(r), t+1)\) and \((\bar{g}_R'(r), t) \rightarrow (\bar{g}_R'(r), t+1)\) if there exists \( k \in K_0 \) such that \( t \geq t_k^a \), and there is one waiting arc \((\bar{g}_R''(r), t) \rightarrow (\bar{g}_R''(r), t+1)\) if there exists \( k \in K_1 \) such that \( t \geq t_k^a \). For each \( k \in K \), if \( k \in K_0 \) and \( t > t_k^a \), then \( \xi_k^{(\bar{g}_R(r), t), (\bar{g}_R(r), t+1)} = \xi_k^{(\bar{g}_R'(r), t), (\bar{g}_R'(r), t+1)} = C_k \); otherwise, \( \xi_k^{(\bar{g}_R(r), t), (\bar{g}_R(r), t+1)} = \xi_k^{(\bar{g}_R'(r), t), (\bar{g}_R'(r), t+1)} = +\infty \). The arc \((\bar{g}_R(r), t) \rightarrow (\bar{g}_R(r), t+1)\) and arc \((\bar{g}_R'(r), t) \rightarrow (\bar{g}_R'(r), t+1)\) allow a short train to park at repair track \( r \)'s position I and II, respectively, when traveling in the outward direction. For each \( k \in K \), if \( k \in K_1 \), \( t \geq t_k^a \), then \( \xi_k^{(\bar{g}_R''(r), t), (\bar{g}_R''(r), t+1)} = C_k \); otherwise, \( \xi_k^{(\bar{g}_R''(r), t), (\bar{g}_R''(r), t+1)} = +\infty \). The arc \((\bar{g}_R''(r), t) \rightarrow (\bar{g}_R''(r), t+1)\) allows a long train to park on the entire repair track \( r \) when traveling in the outward direction.

- **Departure arcs:** For each parking track \( s \in S \) and time instant \( t = 0, 1, \ldots, T \), there are two departure arcs \((\bar{p}_S(s), t) \rightarrow (\bar{p}_S(s), t)\) and \((\bar{p}_S(s), t) \rightarrow (\bar{p}_S(s), t)\) if there exists \( k \in K_0 \) such that \( t \geq t_k^a \), and there are two departure arcs \((\bar{p}_S^0(s), t) \rightarrow (\bar{p}_S(s), t)\) and \((\bar{p}_S^0(s), t) \rightarrow (\bar{p}_S(s), t)\) if there exists \( k \in K_1 \) such that \( t \geq t_k^a \). For each \( k \in K_0 \), if \( t > t_k^a \), then \( \xi_k^{(\bar{p}_S(s), t), (\bar{p}_S(s), t)} = \xi_k^{(\bar{p}_S^0(s), t), (\bar{p}_S(s), t)} = 0 \); otherwise, \( \xi_k^{(\bar{p}_S^0(s), t), (\bar{p}_S(s), t)} = \xi_k^{(\bar{p}_S(s), t), (\bar{p}_S(s), t)} = +\infty \). Arcs \((\bar{p}_S(s), t) \rightarrow (\bar{p}_S(s), t)\) and \((\bar{p}_S(s), t) \rightarrow (\bar{p}_S(s), t)\) represent the situation in which a short train is about to leave the parking track \( s \) when traveling in the inward and outward directions, respectively. For each \( k \in K_1 \), if \( t > t_k^a \), then \( \xi_k^{(\bar{p}_S^0(s), t), (\bar{p}_S(s), t)} = \xi_k^{(\bar{p}_S^0(s), t), (\bar{p}_S(s), t)} = 0 \); otherwise, \( \xi_k^{(\bar{p}_S^0(s), t), (\bar{p}_S(s), t)} = \xi_k^{(\bar{p}_S(s), t), (\bar{p}_S(s), t)} = +\infty \). Arcs \((\bar{p}_S^0(s), t) \rightarrow (\bar{p}_S(s), t)\) and \((\bar{p}_S^0(s), t) \rightarrow (\bar{p}_S(s), t)\) represent the situation in which a long train is about to leave the parking track \( s \) when traveling in the inward and outward directions, respectively.

For each repair track \( r \in R \) and time instant \( t = 0, 1, \ldots, T \), there is a departure arc \((\bar{g}_R(r), t) \rightarrow (\bar{g}_R(r), t)\) if there exists \( k \in K_0 \) such that \( t \geq t_k^a \), and there is a departure arc \((\bar{g}_R''(r), t) \rightarrow (\bar{g}_R''(r), t)\) if there exists \( k \in K_1 \) such that \( t \geq t_k^a \). For each \( k \in K_0 \), if \( t > t_k^a \), then \( \xi_k^{(\bar{g}_R(r), t), (\bar{g}_R(r), t)} = 0 \); otherwise, \( \xi_k^{(\bar{g}_R(r), t), (\bar{g}_R(r), t)} = +\infty \). For each \( k \in K_1 \), if \( t > t_k^a \), then \( \xi_k^{(\bar{g}_R''(r), t), (\bar{g}_R(r), t)} = 0 \); otherwise, \( \xi_k^{(\bar{g}_R''(r), t), (\bar{g}_R(r), t)} = +\infty \). The arc \((\bar{g}_R(r), t) \rightarrow (\bar{g}_R(r), t)\) represents the situation in which a short train has been repaired on repair track \( r \) and is about to leave this repair track. The arc \((\bar{g}_R''(r), t) \rightarrow (\bar{g}_R(r), t)\) represents the situation in which a long train has been repaired at repair track \( r \) and is about to leave this repair track.

- **Transfer arcs:** For each parking track \( s \in S \) and time instant \( t = 0, 1, \ldots, T \), there are four transfer arcs \((\bar{p}(i), t) \rightarrow (\bar{p}_S(s), t), (\bar{p}_S(s), t) \rightarrow (\bar{p}(i'), t), (\bar{p}(i'), t) \rightarrow (\bar{p}_S(s), t), \) and \((\bar{p}_S(s), t) \rightarrow (\bar{p}(i), t)\) if there exists \( k \in K \) such that \( t \geq t_k^a \) and nodes \( i \) and \( i' \) are the endpoints of parking track \( s \).

For each repair track \( r \in R \) and time instant \( t = 0, 1, \ldots, T \), there are also two transfer arcs \((\bar{p}(i), t) \rightarrow (\bar{p}_R(r), t)\) and \((\bar{p}_R(r), t) \rightarrow (\bar{p}(i), t)\) if there exists \( k \in K \) such that \( t \geq t_k^a \) and nodes \( i \) is the endpoint of repair track \( r \).

For each cleaning track \( w \in W \) and time instant \( t = 0, 1, \ldots, T \), there are six transfer arcs \((\bar{p}(i), t) \rightarrow (\bar{p}_W(w), t), (\bar{p}_W(w), t) \rightarrow (\bar{p}(i'), t), (\bar{p}(i'), t) \rightarrow (\bar{p}_W(w), t), \) and \((\bar{p}_W(w), t) \rightarrow (\bar{p}(i'), t)\) if there exists \( k \in K \) such that \( t \geq t_k^a \) and nodes \( i \) and \( i' \) are the endpoints of cleaning track \( w \). These transfer arcs allow a train to traverse the nodes between operation tracks and...
slunting tracks. For each train \( k \in K \), \( \xi^k_{uv} = 0 \) if arc \( u \to v \in A \) is a transfer arc.

- **Switch arcs:** For each repair track \( r \in R \), there are two switch arcs \((\bar{p}_R(r), t) \to (\bar{a}_R(r), t)\) and \((\bar{p}'_R(r), t) \to (\bar{a}'_R(r), t)\) if there exists \( k \in K_0 \) such that \( t \geq t^n_k \), and there is one switch arc \((\bar{p}'_R(r), t) \to (\bar{a}'_R(r), t)\) if there exists \( k \in K_1 \) such that \( t \geq t^n_k \). These arcs represent the situation in which a train has been repaired on track \( r \) at time \( t \) and switches its travel from the inward direction to the outward direction. Note that these arcs change “layer” components from the inward layer to the outward layer. For each track \( k \), there exists \( \xi \) such that \( \xi^k_{(\bar{p}_R(r), t),(\bar{a}_R(r), t)} = 0 \); otherwise, \( \xi^k_{(\bar{p}_R(r), t),(\bar{a}_R(r), t)} = +\infty \). For each train \( k \in K_1 \), if \( t \geq t^n_k \), then \( \xi^k_{(\bar{p}_R(r), t),(\bar{a}_R(r), t)} = 0 \); otherwise, \( \xi^k_{(\bar{p}_R(r), t),(\bar{a}_R(r), t)} = +\infty \).

- **Dummy satisfied-demand arc:** There is a dummy satisfied-demand arc \( o \to d \). One can also say that a train traversing this arc represents the situation in which this train’s daily maintenance requirement is rejected by the depot. A high penalty \( \pi_k \) is incurred if train \( k \) traverses arc \( o \to d \); then we have \( \xi^k_{od} = \pi_k \).

### Appendix B. Proof of Proposition 1

As mentioned in Section 3.2, a path from vertex \( o \) to vertex \( d \) in our two-layer time-space network corresponds to a schedule of a train in the feasible solution. Clearly, a train path in the two-layer time-space network takes either the form \( o \to d \) or the form \( o \to (\rho(i), t) \to \cdots \to (\rho(j), t') \to d \). For each train \( k \in K \), let \( \hat{A}_k \) be the set of the arcs along train \( k \)’s path in a feasible solution of problem \( P \). If train \( k \)’s path takes the form \( o \to d \) (i.e., \( \hat{A}_k = \{o \to d\} \)), we can claim that train \( k \) is rejected by the depot (see Section 3.1.3). That is, train \( k \) is cancelled, which indicates that the repair operation requirement for train \( k \) can be omitted. However because only switch arcs can connect both the in-layer time-space network and the out-layer time-space network, if train \( k \)’s path takes the form \( o \to (\rho(i), t) \to \cdots \to (\rho(j), t') \to d \) rather than the form \( o \to d \), there must be a switch arc in set \( \hat{A}_k \); otherwise, train \( k \)’s path must be infeasible. Note that a switch arc’s preceding arcs include only repairing arcs. To consider the flow balance constraints, before train \( k \) traverses a switch arc, it must traverse a repairing arc along its feasible path, which indicates that train \( k \)'s repair operation requirement is satisfied. This completes the proof of the proposition.  ■
Appendix C. Depot Networks

Figure A1: Network 1.

Figure A2: Network 2.
Figure A3: A depot schedule obtained by Lagr. RSH for an instance in S4.