Simultaneous train rerouting and rescheduling on an N-track network: A model reformulation with network-based cumulative flow variables

Lingyun Meng
Assistant Professor
State Key Laboratory of Rail Traffic Control and Safety
Beijing Jiaotong University
1201, SiYuan Building, No.3 ShangYuanCun
HaiDian District, Beijing 100044, China
Tel.: (86)-10-51688547
Email: lymeng@bjtu.edu.cn

Xuesong Zhou
(Corresponding Author)
Associate Professor
School of Sustainable Engineering and the Built Environment
Arizona State University
Tempe, AZ 85287, USA
Tel.: (1)-480-9655827
Email: xzhou74@asu.edu

Submitted for publication in Transportation Research Part B
First submission: July 31, 2013
First revised version: Dec 24, 2013
Second revised version: Feb 25, 2014

Abstract

Train dispatching is critical for the punctuality and reliability of rail operations, especially for a complex rail network. This paper develops innovative integer programming models for the problem of train dispatching on an N-track network by means of simultaneously rerouting and rescheduling trains. Based on a time-space network modeling framework, we first adapt a commonly used big-M method to represent complex “if-then” conditions for train safety headways in a multi-track context. The track occupancy consideration on typical single and double tracks is then reformulated using a vector of cumulative flow variables. This reformulation technique can provide an efficient decomposition mechanism through modelling track capacities as side constraints which are further dualized through a proposed Lagrangian relaxation solution framework. We further decompose the original complex rerouting and rescheduling problem into a sequence of single train optimization subproblems. For each subproblem, a standard label correcting algorithm is embedded for finding the time dependent least cost path on a time-space network. The resulting dual solutions can be transformed to feasible solutions through priority rules. We present a set of numerical experiments to demonstrate the system-wide performance benefits of simultaneous train rerouting and rescheduling, compared to commonly-used sequential train rerouting and rescheduling approaches.

Keywords Train dispatching, Rail network, Cumulative flow variable, Lagrangian relaxation
1. Introduction

Providing punctual and reliable services is a primary goal of rail industries in order to maintain and further improve their competitive advantages in the rapidly changing multimodal transportation market. Train timetabling typically aims to construct a schedule that specifies a physical network route and detailed arrival time and departure times for each train at passing stations, with the objective to minimize e.g. total travel time of all trains. As tactical plans of complex rail operations, train schedules are programmed and updated every year or every season.

During the process of executing a planned train schedule, various sources of perturbations may influence train running times and dwelling and departing events, thus causing primary delays to the planned train schedule. Due to the high interdependency between trains, primary delays could propagate as secondary delays to other trains on a network. The key task of train dispatching is to take proper measures that can recover impacted schedules from stochastic perturbations and further minimize potential negative consequences. On a high-density rail network with limited capacity, the real-time train dispatching task becomes extremely complicated where ineffective schedule adjustment and ad-hoc responses could significantly downgrade the punctuality and reliability of train services and the overall system performance.

The train timetabling and dispatching problems have been well studied in the past few decades. Assad (1980) and Cordeau et al. (1998) reviewed many key modeling aspects of rail operations and a recent survey by Hansen (2010), summarized emerging methods and solution techniques for train timetabling and dispatching. As real-time train dispatching often relies on planned schedules from the stage of train timetabling, we first start with a brief review on the related train timetabling problem. Carey and Lockwood (1995) presented a mixed integer programming model and solution algorithms for the train timetabling problem on a double-track rail line with trains operating at different speeds. Carey (1994a) further developed an extended model to consider more general and more complex rail networks with possible choices of lines and station platforms. A companion paper by Carey (1994b) proposed an extension from one-way to two-way rail lines. An alternative discrete event simulation-based modeling framework was proposed by Dorfman and Medanic (2004) to solve train scheduling problems on a large-scale rail network. Along this line, Li et al. (2008) presented an improved simulation-based method with a consideration of global train information.

Recently, how to use theoretically sound formulations to describe train timetabling on an N-track network has received considerable attention. Carey and Crawford (2007) developed efficient heuristic algorithms for finding and resolving train conflicts in draft schedules in which the choices of lines/tracks between stations were considered. Lee and Chen (2009) proposed an optimization-oriented heuristic to assign routes and tracks and accordingly allocate time slots for trains. Mu and Dessouky (2011) explored an novel optimization-based procedure for solving the freight train scheduling problem on an N-track network in the context of a rail network in United States, which includes (1) a “FixedPath” model that considers train routing and scheduling sequentially with one exact path for each train, and (2) a “FlexiblePath” model that aims to jointly determine train routing and scheduling decisions based on simplifying assumptions about the topological structure of a network. We will offer more detailed comparisons with their work at the end of this paper. Yan and Yang (2012) proposed a train routing and scheduling model which applies network flow balance constraints and an if-then type of constraints to model rail traffic on a network and safety headways. The if-then type constraint can be handled through a disjunctive graph modeling approach or a big-M type representation in an integer
programming framework. Harrod (2011) proposed a directed hypergraph formulation which is capable to flexibly model both constrained resources and resource transitions by block occupancy side constraints and network transition constraints. By extending optimization models proposed by Zhou and Zhong (2007) and Törnquist and Persson (2007), Castillo et al. (2011) further incorporated user preference into the train timetabling problem.

We now move to a brief review on the train dispatching problem on a rail network. Jovanovic (1989) introduced a mixed integer programming model that minimizes the tardiness cost. Törnquist and Persson (2007) proposed an optimization approach and various practically useful solution strategies for managing railway traffic on an N-track network. Mannino and Mascis (2009) developed a real-time automated traffic control system to operate trains on a network of metro stations in Italy. Iqbal et al. (2013) presented and evaluated three different search parallelization strategies for designing a greedy train dispatching algorithm. By using the multi-strategy based parallel approach, their greedy algorithm shows its advantages in delivering feasible solutions quickly. Meng and Zhou (2011) considered the impact of the stochasticity and dynamicity of perturbations on scheduling robust meet-pass plans by developing a scenario-based rolling horizon solution approach. Quaglietta et al. (2013) proposed an innovative simulation-based optimization framework for analyzing the stability of train dispatching plans under a stochastic and dynamic environment. For more information about the train dispatching problem, we refer to a recent review by Pacciarelli (2013).

Based on a systematic modeling framework with key components such as the alternative graph formulation, branch-and-bound solution algorithms and heuristic real-time conflict-detection-and-resolving algorithms, D’Ariano et al. (2007a, 2007b, 2008, 2009) and Corman et al. (2009, 2010a, 2010b, 2011) proposed a stream of research that addresses various aspects of the train dispatching problem with both small disturbances and heavy disruptions. A laboratory on-line decision support computer system, namely ROMA, was designed and implemented for assisting train dispatchers to recover impacted schedules from perturbations. As indicated by Corman et al. (2010b), compared to conventional solution approaches assuming fixed train routes, better solution quality could be obtained if train routes are allowed to be optimized in conjunction with the change of train orders. Focusing on the problem of managing railway traffic on a large-scale network, Corman et al. (2012, 2014) proposed optimization models and algorithms for coordinating several dispatchers with the objective to drive their behaviors towards globally optimal solutions. Kecman et al. (2013) put forward a set of macroscopic train dispatching models for controlling country-wide railway traffic.

In this research, we adapt the terminology developed by Hansen (2010) and consider the following four typical train dispatching measures: 1) re-time in terms of changing arrival and departure times; 2) re-order in terms of changing arrival and/or departure orders; 3) re-track in terms of using a different track; and 4) re-route in terms of using a different route on a network. Tables 1 and 2 compare recent studies on the train timetabling and train dispatching problems, respectively, in the key dimensions of scheduling strategy, scheduling measure, mathematical formulation, problem size and solution algorithm. Interestingly, we can find that most existing literature sequentially determines routes and then schedules for trains, and there is a clear trend towards the development of detailed mathematical models and effective solution algorithms that can simultaneously (re-)route and (re-)schedule trains. Essentially, there are two categories of modeling approaches for handling time constraints. The first category represents the optimization time horizon in a continuous fashion, and train sequencing constraints are typically used to ensure safety time headways between a pair of train arrival/departure events occurring at any time of the entire planning horizon. On the other hand, many
studies using the Lagrangian relaxation modeling framework (e.g., Brännlund et al., 1998, Caprara et al., 2002) typically discretize the time period of interest to a vector of time intervals (e.g. 1 minute), which are further considered as discrete resources in the dualized problems.

Table 1
Mathematical formulations and solution algorithms for train timetabling.

<table>
<thead>
<tr>
<th>Publication</th>
<th>Scheduling strategy</th>
<th>Scheduling measure</th>
<th>Model structure</th>
<th>Objective</th>
<th>Problem size</th>
<th>Solution algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caprara et al. (2002)</td>
<td>E</td>
<td>T, O</td>
<td>IP</td>
<td>Maximize the sum of profits of the scheduled trains</td>
<td>Up to 500 trains, a rail line of 39 stations</td>
<td>LR, H (Priority rule-based)</td>
</tr>
<tr>
<td>Dorfman and Medanic (2004)</td>
<td>E</td>
<td>T, O</td>
<td>S</td>
<td>Minimize the time to clear the line, delay of all trains, and maximum delay or energy cost</td>
<td>36 trains, 31 sections and 31 nodes</td>
<td>H (Greedy train advance strategy)</td>
</tr>
<tr>
<td>Carey and Crawford (2007)</td>
<td>E</td>
<td>T, O, K</td>
<td>-</td>
<td>Minimize the total cost of scheduled train delays</td>
<td>490 trains, a set of stations with complex track layout</td>
<td>H</td>
</tr>
<tr>
<td>Cacchiani et al. (2008)</td>
<td>E</td>
<td>T, O</td>
<td>IP</td>
<td>Maximize the sum of the profits of the scheduled trains</td>
<td>Up to 221 trains, 102 stations</td>
<td>G, H, BCP</td>
</tr>
<tr>
<td>Lee and Chen (2009)</td>
<td>E</td>
<td>T, O, K, R</td>
<td>MIP</td>
<td>Generate timetables as close to the draft timetable as possible</td>
<td>Up to 128 train services, a single-track rail line of 159.3km and 29 stations, a double-track rail line of 345km and 8 stations</td>
<td>H (Four step process)</td>
</tr>
<tr>
<td>Cacchiani et al. (2010a)</td>
<td>E</td>
<td>T, O</td>
<td>IP</td>
<td>Maximize the profits of all the selected compatible train paths</td>
<td>Up to 210 trains</td>
<td>D, H (Greedy algorithms)</td>
</tr>
<tr>
<td>Cacchiani et al. (2010b)</td>
<td>I</td>
<td>T, O, R</td>
<td>IP</td>
<td>Introduce as many new freight trains as possible</td>
<td>A rail line with up to 69 stations</td>
<td>D, H (Priority rule-based), LR</td>
</tr>
<tr>
<td>Castillo et al. (2011)</td>
<td>E</td>
<td>T, O</td>
<td>MIP</td>
<td>Minimize the sum of the relative travel times</td>
<td>Up to 170 trains</td>
<td>B</td>
</tr>
<tr>
<td>Publication</td>
<td>Scheduling strategy</td>
<td>Scheduling measure</td>
<td>Model structure</td>
<td>Objective</td>
<td>Problem size</td>
<td>Solution algorithm</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>---------------------</td>
<td>--------------------</td>
<td>-----------------</td>
<td>---------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>Mu and Dessouky (2011)</td>
<td>E, I</td>
<td>T, O, K</td>
<td>MIP</td>
<td>Minimize the total delay of all the trains</td>
<td>Up to 60 trains on a medium network, 40 trains on a large network</td>
<td>H (Look-ahead greedy and global neighborhood search)</td>
</tr>
<tr>
<td>Table 2</td>
<td></td>
<td></td>
<td></td>
<td>Mathematical formulations and solution algorithms for train dispatching.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adenso-Diaz et al. (1999)</td>
<td>E</td>
<td>R-T, R-O</td>
<td>MIP</td>
<td>Maximize the number of passenger transported</td>
<td>15 train units, a network of 151.7 km in length</td>
<td>H</td>
</tr>
<tr>
<td>Törnquist and Persson (2007)</td>
<td>E, I</td>
<td>R-T, R-O, R-K, R-R</td>
<td>MIP</td>
<td>Minimize the total final delay of the traffic or minimize the total delay costs</td>
<td>80 trains, a network of 253 segments</td>
<td>H (Four efficient rescheduling strategies)</td>
</tr>
<tr>
<td>D’Ariano et al. (2007a)</td>
<td>E</td>
<td>R-T, R-O</td>
<td>MIP</td>
<td>Minimize the maximum secondary delay for all trains at all visited stations</td>
<td>54 trains circulating each hour, A network of 20km in length, 86 block sections, 16 platforms</td>
<td>B&amp;B, H (FCFS, FLFS)</td>
</tr>
<tr>
<td>D’Ariano et al. (2008)</td>
<td>E</td>
<td>R-T, R-O, R-K, R-R</td>
<td>MIP</td>
<td>Minimize the maximum and average consecutive delays in lexicographic order</td>
<td>52 trains circulating each hour, A network of 50km in length, 191 block sections, 20 platforms</td>
<td>B&amp;B, LS, PR</td>
</tr>
<tr>
<td>Corman et al. (2009)</td>
<td>E</td>
<td>R-T, R-O</td>
<td>MIP</td>
<td>Minimize the (maximum) consecutive delay of all trains and (total) energy consumption</td>
<td>Up to 40 trains per hour, a line of 50km in length, 191 block sections, 21 platforms</td>
<td>B&amp;B, H (FIFO), PR</td>
</tr>
</tbody>
</table>
Corman et al. (2010a)  
R-T, R-O, R-K, R-R  
S  
Multiple criteria such as minimize the total passengers’ delays  
AG

Corman et al. (2010b)  
R-T, R-O, R-K, R-R  
MIP  
Minimize the maximum and average consecutive delays in lexicographic order  
B&B, H (Tabu search)

Corman et al. (2011)  
R-T, R-O  
MIP  
Minimize train delays along other multiple objectives  
B&B, H (Priority rule-based)

Meng and Zhou (2013)  
R-T, R-O, R-K, R-R  
IP  
Minimize the total completion time of all involved trains  
LR, H (Priority rule-based)

Symbol descriptions for Tables 1 and 2:  
Scheduling strategy: Simultaneous (I): determine train route and arrival/departure times of trains / Sequential (E): first specify train routes and then determine the arrival/departure times of each train.  
Scheduling measure: (Re-)time ((R-)T) / (Re-)order ((R-)O) / (Re-)track ((R-)K), typically known as local re-route which means using a different parallel track on the line or within a station / (Re-)route ((R-)R), also known as global re-route which represents selecting an alternative set of segments and stations from origin to destination.  
Model structure: (Mixed) Integer programming (M) IP / Simulation-based model (S).  
Solution algorithm: Branch-and-bound (B&B) / Branch-and-Cut-and-Price (BCP) / Alternative graphs (AG) / Lagrangian relaxation (LR) / Heuristics (H) / Dynamic programming (D) / Local search (LS) / Practical rules (PR) / Bisection method (B) / Column generation (G).

This paper focuses on the train dispatching problem on an N-track network, with the main challenge of how to formulate specific retiming, reordering, retracking and rerouting options in combination. We try to offer the following contributions to the growing body of research work on train dispatching models:

(1) This paper proposes a set of rigorously defined optimization models for the N-track simultaneous train rerouting and rescheduling problem, in which the routes and passing times at each station along the selected route of each train are jointly optimized. Both proposed big-M-based and
cumulative flow variables-based models allow us to fully exploit available station and track capacity in different infrastructure components of a rail network.

(2) Based on a time-space representation for a multi-track network, we first introduce different methods to represent the critical safety time headway for trains that might come from different types of tracks (single-track vs. double-track) with a large number of possible routes through the network. By reformulating the infrastructure capacity implicitly, we then propose a problem decomposition mechanism where each train-specific subproblem only has a smaller variable space compared to the original simultaneous rerouting and rescheduling model involving multiple trains.

(3) By fully taking advantage of an efficient time-dependent shortest path algorithm on a time-space network, we are able to solve the dualized problem rapidly for a medium-size network and further provide better lower bound solutions within a Lagrangian relaxation solution framework. Feasible solutions can be obtained through solution adjustment by priority rules, so the gaps between the lower bound and upper bound of generated solutions offer very useful solution quality measures.

To evaluate the benefits of the simultaneous train rerouting and retiming approach, we also develop a sequential train rerouting and rescheduling model which sequentially determines train routes and then the schedules. The simultaneous approach is proved by experiments to be able to provide better dispatching solutions compared to the sequential approach.

The remainder of this paper proceeds as follows. In Section 2, a conceptual illustration is presented for the simultaneous train rerouting and rescheduling problem on an N-track network. Section 3 develops an optimization model through the commonly used big-M method. Section 4 further constructs a reformulated model based on network cumulative flow variables. Based on a time-space network structure, we use a computationally efficient Lagrangian relaxation solution framework to solve large scale problems in Section 5. The efficiency and effectiveness of proposed models and algorithms, and the benefits of the proposed simultaneous approach are systematically evaluated in Section 6. Finally, concluding remarks and future research extensions are discussed in Section 7, followed by an appendix that formulates the sequential train rerouting and rescheduling model based on cumulative flow variables.

2. Conceptual illustration

This section is intended to conceptually illustrate the two key problems of simultaneous and sequential train rerouting and rescheduling, followed by a problem statement and notation used in the proposed models. In our models, we assume that:

(1) The length of a train is assumed to be zero (as a virtual dot) for simplicity. That is, our model has a macroscopic representation from train characteristics point of view.

(2) The segment between stations is modelled as a series of block sections for a unidirectional double-track rail line and as one block section for a bidirectional single-track line. A block section is denoted as a cell in our paper.

(3) If the constraints on siding tracks are not explicitly considered, a station can be represented as a node; otherwise a station can be considered as a subnetwork with a number of siding tracks being modelled as a set of cells.

(4) Only one train is permitted on a cell at any given time.

(5) The granularity of time is one minute, or a shorter time interval (if required) for some rail systems.
2.1 Simultaneous and sequential train routing and scheduling on a rail network

Fig. 1 depicts a simple rail network with 10 nodes. Nodes 0, 1, 4, 5, 8 and 9 represent a station without a siding track. The node pairs (2,3) and (6,7) respectively correspond to two stations, and in each station there is only one siding track. Each link connects two nodes and it is bidirectional so trains can traverse on both directions. There are four trains to be dispatched through this network, trains 1 and 3 traversing from node 0 to 9, and trains 2 and 4 from node 9 to 0. These four trains have their earliest departure times at minute 0, 15, 30 and 45, respectively. As shown in Fig. 2, there are a total of four possible bidirectional routes for each train, and the corresponding free-flow travel times are 34, 32, 36 and 34 minutes, respectively. A route in this paper can be denoted as a series of nodes. For example, route 1 corresponds to two directional routes, namely node sequences 0-1-4-5-6-7-8-9 and 9-8-7-6-5-4-1-0.

![Fig. 1 A simple rail network with 10 nodes.](image1)

![Fig. 2 Four bidirectional routes that traverse the rail network.](image2)

The main decision variables of train dispatching are (i) routes of trains and (ii) arrival and departure times at each node. In practice, the above optimization model is solved in a sequential fashion. First, we determine the route of each train, for example, all four trains use the route with the minimum free-flow travel time (i.e. route 2) in Fig. 2. Second, trains’ orders and passing times at each node are computed to avoid conflicts. One possible drawback associated with this sequential routing and scheduling method is that the limited options given in the first routing stage could dramatically downgrade the performance of the second “ordering + timing” solutions.

In this paper, we are mainly interested in constructing joint optimization models for both routing and “ordering + timing” tasks. This simultaneous train routing and scheduling approach can serve as a theoretically rigorous benchmark for evaluating the performance of the above-mentioned practical sequential scheduling approach with a predetermined and restrictive route set. From a system-optimal perspective, it is desirable to exploit all possible routes and fully use capacity on all links, for example, links 2-3 and 6-7 with residual capacity but on alternative longer routes.
2.2 Problem statement and notation

Given a network of rail stations and segments with different numbers of tracks, and a set of trains from pre-specified origin stations to destination stations, the N-track train dispatching problem needs to determine the routes and arrival/departure times at each station for a set of trains \( f \in F \), for a given planning horizon \( t = 1, \ldots, T \). \( T \) is the length of planning horizon (e.g., \( T = 300 \) for a 5-hour horizon). A finer discretization (e.g. 30 seconds, 15 seconds) is also possible but could lead to a much larger variable space (Brännlund et al., 1998, Caprara et al., 2002, 2006). In this paper, we represent an original station-segment rail network as \( G = (N, E) \) with a set of nodes \( N \) and a set of cells \( E \). The extended rail network structure in terms of nodes and cells can help us better capture practical safety operational rules, through signaling and interlocking methods. Specifically, a physical double-track segment between stations can be decomposed into a sequence of cells where each cell can correspond to a (directed) physical track circuit. Interested readers can find detailed information on how track circuits are used in Theeg and Vlasenko (2009). Additionally, a station is represented as a single or multiple siding tracks, and each siding track is modeled as a cell in the proposed model. By doing so, we can map the station minimal and maximal dwell time as constraints on traveling time of the corresponding cell(s). Accordingly, a route in this paper is defined as a sequence of cells.

For each cell, the other given input data include its free-flow running time for different types of trains, safety time headways, as well as dwell time requirements. The following parameters for each train are assumed to be given: its origin, destination and earliest departure time. To describe detailed modeling methods, the general subscripts and input parameters of the proposed formulations are first introduced in Tables 3 and 4.

Table 3
General subscripts.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( i, j, k )</td>
<td>node index, ( i, j, k \in N ), ( N ) is the set of nodes</td>
</tr>
<tr>
<td>( e )</td>
<td>cell index, denoted by ( (i, j) ), ( e \in E ), ( E ) is the set of cells</td>
</tr>
<tr>
<td>( p )</td>
<td>route index, ( p \subset P ), ( P ) is the set of all routes on a rail network</td>
</tr>
<tr>
<td>( m )</td>
<td>cell sequence number along a route ( p ), ( m \leq N_p ), ( N_p ) is the number of cells in route ( p )</td>
</tr>
<tr>
<td>( t )</td>
<td>scheduling time index, ( t = 1, \ldots, T ), ( T ) is the planning horizon</td>
</tr>
<tr>
<td>( f )</td>
<td>train index, ( f \in F ), ( F ) is the set of trains</td>
</tr>
</tbody>
</table>

Table 4
Input parameters.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( E_p )</td>
<td>set of sequenced cells of route ( p ), (</td>
</tr>
<tr>
<td>( P_f )</td>
<td>set of possible routes on which train ( f ) may run, ( P_f \subset P )</td>
</tr>
<tr>
<td>( E_f )</td>
<td>set of cells train ( f ) may use, ( E_f \subset E )</td>
</tr>
<tr>
<td>( FT_f(i, j) )</td>
<td>free-flow running time for train ( f ) to drive through cell ( (i, j) )</td>
</tr>
<tr>
<td>( EST_f )</td>
<td>predetermined earliest starting time of train ( f ) at its origin node</td>
</tr>
<tr>
<td>( w_f^{\text{min}}(i, j) )</td>
<td>minimum dwell (waiting) time for train ( f ) on cell ( (i, j) ), ( (i, j) \in \Omega )</td>
</tr>
<tr>
<td>( w_f^{\text{max}}(i, j) )</td>
<td>maximum dwell (waiting) time for train ( f ) on cell ( (i, j) ), ( (i, j) \in \Omega )</td>
</tr>
<tr>
<td>( \text{Cap}(i, j, t) )</td>
<td>flow capacity on cell ( (i, j) ) at time ( t ), ( \text{Cap}(i, j, t) = 0 ) due to maintenance of cell ( (i, j) ) at time ( t ), ( \text{Cap}(i, j, t) = 1 ), otherwise</td>
</tr>
</tbody>
</table>
g  safety time interval between occupancy and arrival of trains
h  safety time interval between departure and release of trains
o_f  origin node of train f
s_f  destination (sink) node of train f
D(s_f)  preferred arrival time at the destination node of train f in the original timetable
\( \theta^p,m_f \)  cell mapping indicator, represents the cell that is the m\textsuperscript{th} cell for train f along route p
\( E_s(i) \)  set of cells starting from or ending at node i
\( E_o(i) \)  set of cells starting from node i
\( E_d(i) \)  set of cells ending at node i
\( \Omega \)  set of cells that allow dwell time, representing siding tracks in stations

3. Simultaneous train rerouting and rescheduling model

Before detailing the Simultaneous train Rerouting and Rescheduling (SRR) model, we first introduce the integral decision variables in Table 5.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_f(i,j)</td>
<td>0-1 binary train routing variables, x_f(i,j)=1, if train f selects cell (i, j) on the network, and otherwise x_f(i,j)=0</td>
</tr>
<tr>
<td>a_f(i,j)</td>
<td>arrival time of train f at cell (i, j)</td>
</tr>
<tr>
<td>d_f(i,j)</td>
<td>departure time of train f from cell (i, j)</td>
</tr>
<tr>
<td>( \theta(f,f',i,j) )</td>
<td>0-1 binary train ordering variables, ( \theta(f,f',i,j)=1 ) if train f' arrives at cell (i, j) after train f, and otherwise ( \theta(f,f',i,j)=0 )</td>
</tr>
<tr>
<td>TT_f(i,j)</td>
<td>running time for train f at cell (i, j)</td>
</tr>
</tbody>
</table>

The vector \( x_f(i,j) \) captures the routing decisions on a rail network. The vectors \( a_f(i,j) \) and \( d_f(i,j) \) have been commonly used in previous studies to directly describe the timestamps when the arrival and departure events of train f at cell (i, j) occur. The vector \( \theta(f,f',i,j) \) represents the train orders, which are typically used for modelling safety time headways.

The objective is to minimize the total deviation time of all involved trains. The total deviation time of train f can be described as \( \sum_i (\sum_d d_f(i,s_f) - D(s_f)) \). Note that this objective function uses the same weight on early arrival deviation and late arrival deviation, while it is possible to have different weights on these two schedule deviation components (INFORMS RAS, 2012). The SRR model is now formulated as the following problem (P1).

\[
(P1) \quad Z = \min \sum_f \left\{ \sum_i (\sum_d d_f(i,s_f) - D(s_f)) \right\}
\]

Subject to

**Group I: Flow balance constraints**

Flow balance constraints at the origin node:
\[
\sum_{i,j \in E^f} x_f(i, j) = 1, \forall f
\]

Flow balance constraints at intermediate nodes:
\[
\sum_{i \in (i, j) \in E^f} x_f(i, j) = \sum_{k \in (j, k) \in E^f} x_f(j, k), \forall f, j \in N - \alpha_j - s_f
\]

Flow balance constraints at the destination node:
\[
\sum_{i \in (i, j) \in E^f} x_f(i, j) = 1, \forall f
\]

**Group II: Time-space network constraints**

Starting time constraints at the origin node:
\[
\sum_{j \in (i, j) \in E^f} \bar{a}_f(i, j) \geq EST_f, \forall f
\]

Cell-to-cell transition constraints:
\[
\sum_{i \in (i, j) \in E^f} \bar{d}_f(i, j) = \sum_{j \in (j, k) \in E^f} \bar{a}_f(j, k), \forall f, j \in N - s_f - e_f
\]

Mapping constraints between time-space network and physical network
\[
x_f(i, j) - 1 \leq \bar{a}_f(i, j), \forall f, (i, j) \in E_f
\]
\[
x_f(i, j) - 1 \leq \bar{d}_f(i, j), \forall f, (i, j) \in E_f
\]

**Group III: Running time and dwell time constraints**

Running time constraints:
\[
TT_f(i, j) = d_f(i, j) - \bar{a}_f(i, j), \forall f, (i, j) \in E_f
\]

Minimum running time constraints:
\[
TT_f(i, j) + (1 - x_f(i, j)) \times M \geq FT_f(i, j), \forall f, (i, j) \in E_f
\]

Minimum and maximum dwell time constraints:
\[
TT_f(i, j) + (1 - x_f(i, j)) \times M \leq w^\text{max}(i, j) + FT_f(i, j), \forall f, (i, j) \in E_f \cap \Omega
\]

**Group IV: Mapping constraints between train orders and cell usage on the same track**
\[
x_f(i, j) + x_f(i, j) - 1 \leq \theta(f, f', i, j) + \theta(f', f, i, j) \leq 3 - x_f(i, j) - x_f(i, j), \forall f, f', f \neq f', (i, j) \in E_f
\]
\[
\theta(f, f', i, j) \leq x_f(i, j), \forall f, f', f \neq f', (i, j) \in E_f
\]
\[
\theta(f, f', i, j) \leq x_f(i, j), \forall f, f', f \neq f', (i, j) \in E_f
\]

**Group V: Capacity constraints on the same track**
\[
\bar{a}_f(i, j) + (3 - x_f(i, j) - x_f(i, j) - \theta(f, f', i, j)) \times M \geq \bar{d}_f(i, j) + g + h, \forall f, f', f \neq f', (i, j) \in E_f
\]
\[
\bar{a}_f(i, j) + (3 - x_f(i, j) - x_f(i, j) - \theta(f, f', i, j)) \times M \geq \bar{d}_f(i, j) + g + h, \forall f, f', f \neq f', (i, j) \in E_f
\]

In Group I, constraints (2), (3) and (4) ensure flow balance on the network at the origin node of train \( f \), intermediate nodes, and the destination node respectively.

In Group II, constraints (5) make sure that trains do not depart earlier than the predetermined earliest starting time at their origin nodes. Constraints (6) aim to guarantee \( \bar{a}_f(j, k) = \bar{d}_f(i, j) \) if the adjacent cells \( (i, j) \) and \( (j, k) \) are both used by train \( f \). Constraints (7) and (8) are imposed to map
the variables $\overline{a}_f(i,j)$ and $\overline{a}_f(i,j)$ in time-space network to the variables $x_f(i,j)$ in physical network, so as to describe whether cell $(i,j)$ is selected by $f$ for traversing the network from its origin to destination.

In Group III, constraints (10) enforce the required minimum running time and constraints (11, 12) guarantee minimum and maximum station dwell times by the variables $TT_f(i,j)$. $TT_f(i,j)$ means the running time for train $f$ on cell $(i,j)$, which can be computed by Eq. (9). It should be noted that the input parameter $w_{ij}^{\text{min}}(i,j)$ is normally set to be equal to the value used in train timetabling stage. For a congested part of a network, it should be a sufficiently large value (e.g. 1 hour) to avoid model infeasibility (Carey and Lockwood, 1995).

In Group IV, constraints (13) link train orders variables $\theta(f',f',i,j)$ and cell usage variables $x_f(i,j)$. More specifically, constraints (13) make sure that, if and only if both trains $f$ and $f'$ traverse on cell $(i,j)$, i.e., $x_f(i,j) = x_{f'}(i,j) = 1$, then both inequalities reduce to $\theta(f,f',i,j) + \theta(f',f,i,j) = 1$. This equality (with two 0-1 binary variables of $\theta(f,f',i,j)$ and $\theta(f',f,i,j)$) further indicates that, either train $f$ arrives at cell $(i,j)$ after train $f'$ or train $f'$ arrives at cell $(i,j)$ after train $f$. If $x_f(i,j) = 0, x_{f'}(i,j) = 1$ or $x_f(i,j) = 1, x_{f'}(i,j) = 0$ or $x_f(i,j) = x_{f'}(i,j) = 0$, then constraints (13) reduces to non-active inequalities, since $\theta(f,f',i,j) + \theta(f',f,i,j)$ is always between 0 and 2. Constraints (14) and (15) further ensure that any $\theta(f',f',i,j)$ and $\theta(f',f',i,j)$ are always less than $x_f(i,j)$ and $x_{f'}(i,j)$. $M$ is a sufficiently large positive number.

In Group V, constraints (16) and (17) explicitly ensure the cell capacity requirement by setting a safety headway which is computed by $g+h$ between the departure of a preceding train and the arrival of a following train. If those two trains are running on the same cell.

Specifically, for train $f$ and $f'$ traversing on cell $(i,j)$, i.e., $x_f(i,j) = x_{f'}(i,j) = 1$, constraints (16, 17) can be reduced to common if-then conditions as discussed below.

1. If train $f'$ arrives at cell $(i,j)$ after train $f$ (i.e., $\theta(f,f',i,j) = 1$), then there should be a safety time headway $g+h$ between the arrival time of train $f$ and the departure time of train $f'$ on cell $(i,j)$, please see constraints (16);

2. If train $f'$ arrives at cell $(i,j)$ after train $f$ (i.e., $\theta(f',f,i,j) = 1$), then there should be a safety time headway $g+h$ between the arrival time of train $f'$ and the departure time of train $f$ on cell $(i,j)$, please see constraints (17).

Moreover, Table 6 enumerates different statuses of constraints (16) and (17) through combined binary indicators $\text{CBI} = 3 - x_f(i,j) - x_{f'}(i,j) - \theta(f,f',i,j)$ and $\text{CBI}' = 3 - x_f(i,j) - x_{f'}(i,j) - \theta(f',f,i,j)$ in more detail.

Table 6

<table>
<thead>
<tr>
<th>Combined variables</th>
<th>$\theta(f,f',i,j) = 0$</th>
<th>$\theta(f,f',i,j) = 1$</th>
<th>$\theta(f',f,i,j) = 1$</th>
<th>$\theta(f',f,i,j) = 0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_f(i,j) = x_{f'}(i,j) = 1$</td>
<td>$x_f(i,j) = 1$</td>
<td>$x_f(i,j) = 1$</td>
<td>$x_{f'}(i,j) = x_f(i,j) = 0$</td>
<td>or $x_{f'}(i,j) = x_f(i,j) = 0$</td>
</tr>
</tbody>
</table>
4. Model reformulation based on network cumulative flow variables

Considering the above big-$M$ type representation for the either-or constraints in the train dispatching problem, many studies are devoted to efficient decomposition mechanisms for reducing the model complexity, and heuristic algorithms for obtaining feasible solutions within reasonable computational time (e.g., train-based decomposition by Carey and Lockwood, 1995, Lee and Chen, 2009). In the SRR model, the coupled capacity constraints and cell usage constraints (13-17) are extremely difficult to decompose into different solution branches, as there are a large number of possible routes leading to very different cell usage combinations.

Caprara et al. (2002, 2006) proposed a method to model track capacity constraints in an implicit fashion through a set of clique inequality constraints, which are further dualized in a Lagrangian relaxation solution framework. Their models are constructed in a single line context while the number of clique constraints can grow exponentially as the number of trains increases. As a result, they adopted a relax-and-cut method to consider a limited set of active constraints. In a rail network, modelling track capacities and safety headway requirements through clique inequality constraints could become even more difficult especially at nodes with double tracks merging into a single track or a single track diverging to double tracks.

Based on network cumulative flow variables, this section aims to propose a REFormulated Simultaneous train Rerouting and Rescheduling model (REF-SRR). In this reformulated model, we can easily model temporal and spatial occupancy of trains on tracks and safety time headway under a multi-track context. Moreover, through the time-space discretization of track occupancy features, the track capacities in a network can be modelled as discretized resources and REF-SRR provides a tractable foundation for efficient train-based decomposition.

We first introduce the decision variables and the key modelling features of spatial occupancy and safety time headway through network cumulative flow variables, and then describe REF-SRR model in detail.

Recall that $x_{ij}(i, j)$ is used to capture the routing decisions on a rail network. Two more sets of variables are introduced in Table 7 to describe the complex relationship between routes and train (time-space) paths. Specifically, vector $y_{ij}(i, j, t)$ describes a detailed train path through the extended time-space network, and cumulative flow variables $a_{ij}(i, j, t)$ and $d_{ij}(i, j, t)$ are used to represent both temporal and spatial resource consumption of trains.

\[
x_{ij}(i, j) = 1,
\]
\[
x_{ij}(i, j) = 0
\]
14

Table 7
Decision variables for the reformulated simultaneous train rerouting and rescheduling model.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_f(i, j, t)$</td>
<td>0-1 binary time-space occupancy variables for time-space network, $y_f(i, j, t) = 1$, if train $f$ occupies cell $(i, j)$ at time $t$, and otherwise $y_f(i, j, t) = 0$</td>
</tr>
<tr>
<td>$a_f(i, j, t)$</td>
<td>0-1 binary cumulative arrival flow variables, $a_f(i, j, t) = 1$ if train $f$ has already arrived at cell $(i, j)$ by time $t$, and otherwise $a_f(i, j, t) = 0$</td>
</tr>
<tr>
<td>$d_f(i, j, t)$</td>
<td>0-1 binary cumulative departure flow variables, $d_f(i, j, t) = 1$, if train $f$ has already departed from cell $(i, j)$ by time $t$, and otherwise $d_f(i, j, t) = 0$</td>
</tr>
</tbody>
</table>

4.1 Modeling spatial occupancy and safety headway through network cumulative flow variables

This section aims to give a detailed discussion on network cumulative flow variable-based modeling techniques for handling spatial occupancy and safety headway constraints. On an $N$-track network, a challenging issue is how to consider temporal and spatial capacity constraints for both single-track and double-track lines. Without loss of generality, the planning horizon is discretized and denoted as integers from time index 1 to $T$.

As illustrated in Fig. 3(a), train $f$ arrives at cell $(i, j)$ at time 8 and departs at time 10. It can be further reformulated through cumulative flow variables $a_f(i, j, t)$ and $d_f(i, j, t)$. As shown in Fig. 3(b), there is a change of $a_f(i, j, t)$ at time 8 and a change of $d_f(i, j, t)$ at time 10, from 0 to 1. The transformation from $a_f(i, j)$ to $a_f(i, j, t)$ can be formally represented by

$$
\overline{a}_f(i, j) = \sum_{t=1}^{T} t \times [a_f(i, j, t) - a_f(i, j, t-1)]
$$

Similarly, we have the following representation for departure variables $\overline{d}_f(i, j)$:

$$
\overline{d}_f(i, j) = \sum_{t=1}^{T} t \times [d_f(i, j, t) - d_f(i, j, t-1)]
$$

![Fig. 3 Reformulation of arrival and departure time variables through cumulative flow variables](image)

We now need to consider how to model spatial occupancy and safety headways. A set of shifted cumulative flow variables $a_f(i, j, t) + g$ and $d_f(i, j, t) - h$ is introduced in this paper to represent whether train $f$ starts or ends occupying cell $e$ by time $t$, by considering minimum safety time headway $g$ and $h$. Finally, we can represent the spatial occupancy of a train through a simple equation.
Let’s assume \( g = h = 1 \), the gray rectangle block in Fig. 4 corresponds to \( y_j(i, j, t) = 1 \) for \( t = 7...10 \), \( y_j(i, j, t) = 0 \) otherwise.

\[
y_j(i, j, t) = a_j(i, j, t+g) - d_j(i, j, t-h).
\]

Fig. 4 Spatial occupancy of cell \((i, j)\) by train \(f\) between time 7 min and 11 min.

We now move to a double-track segment case with four sequencing cells \(e_1, e_2, e_3\) and \(e_4\). As shown in Fig. 4, the stacks of gray rectangles represent the detailed occupancy by trains \(f\) and \(f'\) in a time-space network. On a double-track segment, it is possible that two trains are running at the same time (e.g. at minute 14 in Fig. 5), while there is only one train being allowed on each cell (i.e. track circuit). The proposed reformulation using cumulative flow variables can nicely capture the above two requirements through \( y_j(i, j, t) + y_j(i, j, t) \leq 1 \). More specifically, at minute 14 of Fig. 5, both trains \(f\) and \(f'\) are running at the segment from station A to B, but trains \(f\) and \(f'\) do not occupy any cell in the segment at the same time. For example, trains \(f\) occupies cell \(e_2(i, j)\) in the time span between \(t = 7\) and \(t = 10\), and train \(f'\) occupies \(e_2(i, j)\) between \(t = 13\) and \(t = 15\).

Fig. 5 Cell decomposition of a double-track segment \(l\) between stations A and B.

A single-track case is illustrated in Fig. 6. We need to introduce directed cell \(e\) from station \(i\) to \(j\) and cell \(e'\) from station \(j\) to \(i\), in order to allow trains running on opposite directions. Let us consider train \(f\) using \(e\) and train \(f'\) using \(e'\). Since cells \(e\) and \(e'\) correspond to the same segment, we can use a constraint of \( y_j(i, j, t) + y_j(j, i, t) \leq 1 \) to model the safety headway requirement. More specifically,
\[
y^*_i(j,t) + y^*_j(i,t) = 1 \quad \text{for } t \text{ between 3 and 9, 11 and 16}, \quad y^*_i(j,t) + y^*_j(i,t) = 0 \quad \text{for } t \text{ between 0 and 2, 9 and 11 (2 time units buffer time), 16 and 25}.
\]

In a rail network context in which there are both double-track segments and single-track segments, one needs to apply the above two methods to represent the double/single track segments by cells. Moreover, as pointed by Harrod (2011), on a rail network, deadlock(s) may occur if block/cell occupancy are simply implemented resulting in inflated objective values. Thanks to the introduction of safety headways \( g / h \), which can prevent more than one train arriving at a node at the same time, the proposed cell occupancy representation is capable of avoiding the occurrence of deadlock(s).

Cumulative flow counts-based methods have been widely used in highway traffic engineering. Specifically, cumulative flow counts are typically treated as continuous variables to describe macroscopic characteristics and relationships between different traffic stream measures and vehicular travel times (Hall, 2003; Cassidy, 2003). In comparison, the cumulative flow variables used in this paper are a special vector of 0-1 binary variables that allows us to clearly define (microscopic and train-specific) temporal and spatial capacity constraints in a time-space network. In the area of air traffic flow management, Bertsimas and Stock (1998) introduced 0-1 variables as the summation of flow variables by time \( t \) to construct a series of time-dependent multi-commodity network flow models. Although they have not specifically used the terminology of cumulative flow variables, such formulation facilitates them to model connectivity between sectors along a predetermined flight route. In the train timetabling/dispatching field, to the best of our knowledge, this paper is the first study using cumulative flow variables-based modeling methods that allow simultaneous rerouting and rescheduling decisions.

In our paper, the proposed network cumulative flow variables-based representation enables many unique modeling features. First, it can easily capture complex safety headway constraints in a network with both single and double tracks, with/without a predetermined route, through reformulating the temporal and spatial resource occupancy of trains. Second, it enables a very efficient problem decomposition mechanism by trains, while each subproblem is relatively simple to solve in its extended time-space network.

4.2 Simultaneous train rerouting and rescheduling model based on network cumulative flow variables

The reformulated simultaneous train rerouting and rescheduling model based on network cumulative flow variables is now formulated as the following problem (P2).
\[
Z = \min \sum_{f} \left\{ \sum_{i} \left[ \sum_{t} t \times \sum_{(i,j) \in E_f \setminus \{f\}} \left[ d_f(i,s_f,t) - d_f(i,s_f,t-1) \right] - D(s_f) \right] \right\} 
\]

Subject to

**Group I: Flow balance constraints**

Flow balance constraints at the origin node:
\[
\sum_{i,j \in E(\alpha_f \setminus f)} x_f(i,j) = 1, \forall f
\]

Flow balance constraints at intermediate nodes:
\[
\sum_{i,k \in E(j \setminus f)} x_f(i,k) = \sum_{j,k \in E(j \setminus f)} x_f(j,k), \forall f, j \in N - o_f - s_f
\]

Flow balance constraints at the destination node:
\[
\sum_{i,j \in E(f \setminus \{f\})} x_f(i,j) = 1, \forall f
\]

**Group II: Time-space network constraints**

Starting time constraints at the origin node:
\[
\sum_{j \in E(\alpha_f \setminus f)} a_f(o_f, j, t) = 0, \forall f, t < EST_f
\]

Within cell transition constraints:
\[
d_f(i,j,t + FT_f(i,j)) \leq a_f(i,j,t), \forall f, (i,j) \in E_f, t
\]

Cell-to-cell transition constraints:
\[
\sum_{i,j \in E(f \setminus \{f\})} d_f(i,j,t) = \sum_{j,k \in E(f \setminus \{f\})} a_f(j,k,t), \forall f, j \in N - s_f - e_f, t
\]

Mapping constraints between time-space network and physical network
\[
x_f(i,j) = a_f(i,j,T), \forall f, (i,j) \in E_f
\]

**Group III: Running time and dwell time constraints**

Running time constraints:
\[
TT_f(i,j) = \sum_{t} \left[ t \times \left[ d_f(i,j,t) - d_f(i,j,t-1) \right] \right] - \sum_{t} \left[ t \times \left[ a_f(i,j,t) - a_f(i,j,t-1) \right] \right], \forall f, (i,j) \in E_f
\]

Minimum running time constraints:
\[
TT_f(i,j) \geq FT_f(i,j), \forall f, (i,j) \in E_f
\]

Minimum and maximum dwell time constraints:
\[
w_f^{\min}(i,j) + FT_f(i,j) \leq TT_f(i,j) \leq w_f^{\max}(i,j) + FT_f(i,j), \forall f, (i,j) \in E_f \cap \Omega
\]

**Group IV: Capacity constraints**

Cell occupancy indication constraints:
\[
y_f(i,j,t) = a_f(i,j,t + g) - d_f(i,j,t - h), \forall f, (i,j) \in E_f, t
\]

Cell capacity constraints:
\[
\sum_{f \in E_f} y_f(i,j,t) + \sum_{f \setminus \{i,j\}} y_f(i,j,t) \leq Cap(i,j,t), \forall i, j, t
\]

**Group V: Time-connectivity constraints for cumulative flow variables**
\begin{align}
a_f(i, j, t) &\geq a_f(i, j, t-1), \forall f, (i, j) \in E_f, t \\
d_f(i, j, t) &\geq d_f(i, j, t-1), \forall f, (i, j) \in E_f, t
\end{align}

(34) (35)

In Group I, similar to constraints (2), (3), (4), constraints (21), (22) and (23) ensure flow balance on the network at the origin node of train \( f \), intermediate nodes, and the destination node respectively.

In Group II, constraints (24) and (25) make sure that trains do not depart earlier than predetermined earliest starting time at their origin nodes. Constraints (26) ensure the transition constraints within cells. Constraints (27) aim to guarantee \( a_f(j, k, t) = d_f(i, j, t) \) if the adjacent cells \((i, j)\) and \((j, k)\) are both used by train \( f \). Constraints (28) are imposed to map the variables \( a_f(i, j, t) \) in time-space network to the variables \( x_f(i, j) \) in physical network, so as to describe whether cell \((i, j)\) is selected by \( f \) for traversing the network from its origin to destination.

In Group III, constraints (30) and (31) enforce the required minimum running time as well as minimum and maximum station dwell times by the variables \( TT_f(i, j) \). \( TT_f(i, j) \) means the running time for train \( f \) on cell \((i, j)\), which can be computed by Eq. (29).

In Group IV, constraints (32) link \( y_f(i, j, t) \) with \( a_f(i, j, t+g) \) and \( d_f(i, j, t-h) \). Note that the first term will be 1 if train \( f \) has started occupying cell \((i, j)\) by time \( t \) and the second term will be 1 if train \( f \) has ended occupying cell \((i, j)\) by time \( t \). Therefore, the only trains that contribute a value of 1 to the difference \( a_f(i, j, t+g)-d_f(i, j, t-h) \) represent the trains that are occupying cell \((i, j)\) at time \( t \), i.e., \( y_f(i, j, t) = 1 \). Recall that in Fig. 4, \( y_f(i, j, t) = 1 \) for \( t = 7 \ldots 10 \) and \( y_f(i, j, t) = 0 \) otherwise.

Furthermore, constraints (33) make sure the number of trains that are occupying cell \((i, j)\) is less than the capacity of cell \((i, j)\), which implicitly ensures safety time headways between trains. The additive structure of capacity usage (left portion of constraints (33)) is a key technique of the reformulated model as it can decouple the original problem into many train-specific subproblems. The mechanism is later used by a Lagrangian relaxation solution framework in section 5.

It should be noted that the occupancy of cell from \( j \) to \( i \) should also be counted into the occupancy of cell from \( i \) to \( j \) by train \( f \), and vice versa, as cell \((i, j)\) and \((j, i)\) essentially refer to one physical track circuit.

In Group V, constraints (34) and (35) represent connectivity in time. Thus, if train \( f \) has arrived at or departed from cell \((i, j)\) by time \( t \), then \( a_f(i, j, t) \) or \( d_f(i, j, t) \) has to have a value of 1 for all later time periods, \( t' \geq t \).

Thanks to the proposed modeling approach based on network-wide cumulative flow variables, the above model can easily model spatial occupancy and safety headways and work on a complex network with both unidirectional double tracks and bidirectional single tracks. This relaxes the assumption that “each track can only be traversed in one direction by a train whose traveling direction is given” in Mu and Dessouky (2011). We will discuss how the REF-SRR model, proposed in this paper, is related to the previous work of Mu and Dessouky (2011) in detail in Section 7. In order to evaluate the benefits of the proposed simultaneous train rerouting and rescheduling approach, compared to practically-used sequential rerouting/rescheduling approaches, we also develop a SEQuential train rerouting and rescheduling model (SEQ) based on cumulative flow variables as a benchmark, which is detailed as model (P3) in the appendix.

5. Lagrangian relaxation based solution procedure

Although considerable progress has been made on both linear/integer programming formulations and solution algorithms for train rerouting and rescheduling problems, the existing methods either rely
on commercial optimization solvers to solve the formulated model, which could take a significant amount of running time and memory space to solve a real-world problem to optimality, or many heuristic rules that could efficiently find practically-satisfactory solutions but still with a lack of solution quality assessment.

Brännlund et al. (1998) first introduced a Lagrangian relaxation approach to determine a profit maximizing schedule in which track resources are modeled by additive constraints and then relaxed in a quadratic optimization subproblem. Their resource constraints are relatively simple in a sense that there can be only one train on each block at each time period. Based on a graph-theoretic formulation for the periodic-timetabling problem, Caprara et al. (2002) modeled limited resources by incompatible arcs (i.e. conflicting operations) and forbid the simultaneous selection of such arcs through a novel concept of clique constraints. They presented a Lagrangian relaxation solution method, and many additional practical constraints are incorporated specifically in the study by Caprara et al. (2006). In a single-track corridor, Zhou and Zhong (2007) modeled limited track resources at both segments and stations to consider headway constraints, which are further relaxed through the Lagrangian relaxation technique to compute a lower bound. It should be remarked that, the above Lagrangian relaxation techniques are applied to solve the train routing and scheduling problem in a single or double track rail line with a route being specified a priori for each train. In comparison, our proposed Lagrangian relaxation solution framework further allows trains to select an optimal route (among different feasible routes) and the corresponding schedule that can minimize the deviation time in a complex time-space network.

This section aims to solve the proposed model (P2) which simultaneously reroute and reschedule trains in an N-track network with both unidirectional and bidirectional tracks. We first dualize the complex constraints in (P2) and solve the relaxed problem using a computationally efficient shortest path algorithm in a carefully constructed time-space network representation. The Lagrangian relaxation solution framework used in this research can help to construct a tight lower bound and then provide a good base solution for generating feasible solutions with valid upper bounds. We next detail the overall Lagrangian relaxation solution framework and the underlying label-correcting algorithm for solving the time-dependent least cost path problem, as well as the priority rule-based method for transforming dual solutions to feasible solutions.

5.1 Lagrangian relaxation solution framework

The key modeling aspect need to be discussed in the proposed Lagrangian relaxation framework is which constraints should be relaxed. The constraints in (P2) can be classified into two categories. The first category includes Groups I, II, III and V, and they are all directly related to individual trains. The second category contains cell capacity constraints of Group IV, which is in general difficult to solve as it involves all trains on the same cell. Note that the difficulties of different constraints groups are examined in section 6.

We introduce a set of nonnegative Lagrangian multipliers \( \rho_{i,j,t} \) to dualize the additive cell capacity constraints (33) as a penalty term in the following relaxed model (P4):

\[
\min Z = \sum_{f} \left\{ \sum_{t} \sum_{i,j,s} \left[ d_{f}(i,s,j,t) - d_{f}(i,s,j,t-1) \right] D(s) \right\} + \sum_{i,j,t} \rho_{i,j,t} \left\{ \sum_{f(i,j) \in E_{f}} \left[ y_{f}(i,j,t) \right] + \sum_{f(j,i) \in E_{f}} \left[ y_{f}(j,i,t) \right] - Cap(i,j,t) \right\}
\]
Subject to constraints (21-32) and (34, 35).

The multiplier \( \rho_{i,j,t} \) in Eq. (36) can be interpreted as the cost charged for utilizing the resource (i.e. cell \((i,j)\)) at time \( t \). Essentially, the major goal of the Lagrangian function is to balance the total train deviation time, and the cost for utilizing limited facility resources through paying appropriate resource prices. To maximize the lower bound value of (P2), we need to solve the following Lagrangian dual problem:

\[
\max_{\rho_{i,j,t}} \quad LR = - \sum_{i,j} \sum_{t} \left( \rho_{i,j,t} \times Cap(i,j,t) \right) + \min \sum_{t} LR_f 
\]

where

\[
LR_f = \sum_{t} \sum_{(i,j) \in E} \left[ d_f(i,e_f,t) - d_f(i,e_f,t-1) \right] - D_s f + \sum_{(i,j) \in E} \rho_{i,j,t} \times y_f(i,j,t) 
\]

Clearly, the original interrelated system is separated into a set of subproblems, and the inner minimization problem is concerned with the sum of \( LR_f \) for all trains. In a decomposed subproblem \( LR_f \), the deviation time of train \( f \) (i.e. the first portion of Eq. (38)) is expressed as the deviation time of train \( f \) at its destination node \( s_f \), compared to its preferred arrival time. The resource cost of train \( f \) (i.e. the second portion of Eq. (38)) for traversing a network from the origin node to the destination node is computed by summing \( \rho_{i,j,t} \) over all selected cells within associated time spans.

Given a set of resource prices, we need to find the least cost path of train \( f \) from its origin node to its destination node. As a result, the train-based subproblems are now transformed to a sequence of time-dependent least cost path problems. Those problems seek to find the best resource utilization scheme for each train subject to constraints I, II, III and V, which restrict possible state transitions in the time-space network.

Since the dual cost function (37) is not differentiable everywhere, we solve the dual problem by updating \( \rho_{i,j,t} \) using the standard sub gradient method, which is intended to iteratively adjust the resource prices by setting

\[
\rho_{i,j,t}^{q+1} = \max \left\{ 0, \rho_{i,j,t}^{q} + \alpha^{q} \times \left[ \sum_{f(i,j) \in E_f} \left[y_f(i,j,t)\right] + \sum_{f(j,i) \in E_f} \left[y_f(j,i,t)\right] - Cap(i,j,t) \right] \right\} 
\]

where the superscript \( q \) is the iteration index used in the dual updating procedure, and \( \rho_{i,j,t}^{q} \) and \( \alpha^{q} \) denote the cell multiplier values, and step size at iteration \( q \), respectively. In the optimum search process, the step size parameter is updated as

\[
\alpha^{q} = 1/(q+1) 
\]

and we stop reducing \( \alpha^{q} \) after a certain number of iterations.

It is well known that the applicability and effectiveness of the Lagrangian method rely on having a relatively small number of constraints to dualize. Since there are total \( |E| \) cell resources, the dimension of complete capacity constraints in the simultaneous rerouting and rescheduling problem is extremely large. Recognizing most resource constraints are non-binding in the optimal train dispatching solution, the relax-and-cut logic described in Caprara et al. (2002) and adapted by Zhou and Zhong (2007) is used here to dynamically relax resource capacity constraints by only dualizing a subset of constraints at every iteration. Specifically, if a resource has not been used within recent several iterations, the algorithm automatically resets the price for the unused resource back to zero.
With this dynamic constraint generation scheme, the set of Lagrangian multipliers to be updated varies along the iterative process and the multi-dimensional resource price vector is relatively easy to be stabilized.

The Lagrangian relaxation solution framework is now summarized as follows.

**Input:** Identical to the input of Model (P2).

**Output:** Dual solutions with optimality gap $\varepsilon$ for (P4) and the corresponding feasible solutions for (P2).

**Step 1. (Initialization)**

Let $q = 1$, initialize the multipliers $\lambda_{i,j,t}^q = 0$, initialize the step size $\alpha^q = 0.5$.

**Step 2. (Solve the relaxed model (P4))**

(1) Solve the model (P4) by the time-dependent least cost path algorithm introduced in Section 5.3;
(2) Compute the lower bound of (P4), denoted by LB_P4.

**Step 3. (Transform dual solutions to feasible solutions)**

Use the priority rule-based implementing algorithm introduced in Section 5.4 to transform the dual solutions to feasible solutions, and compute the upper bound of (P4), denoted by UB_P4.

**Step 4. (Compute the optimality Gap)**

Compute the optimality gap $\varepsilon$ between LB_P4 and UB_P4.

**Step 5. (Update Lagrangian multipliers)**

Compute the multipliers for the next iteration by Eq. (39).

**Step 6. (Termination condition)**

If $q > Q_{\text{max}}$ (a predetermined maximum number of iteration) or the gap is less than the predetermined value $\varepsilon^*$, or the upper bound UB_P4 does not improve for a certain number of iterations (e.g. 100), then the algorithm will be terminated. Otherwise, loop to Step 2.

5.2 Extended time-space network for time-dependent shortest path algorithm

In order to compute $\min \sum_f LR_f$ in Eq. (37), we need to solve a set of the least cost path problems through a cell-based network $G = (N, E)$. This network $G$ is then further extended to a time-space network $TSG = (V, A)$ for each train $f$. Each node in set $N$ is extended to a set of vertexes $(i, t)$ in the set of time-space network at each time interval $t$ in the planning horizon, $t = 1, 2, \ldots, T$. To take into account feasible transitions allowed in the $N$-track train dispatching problem under consideration, we need to define three types of arcs carefully.

(1) Cell traveling arcs are extended from a cell $(i, j)$ and each arc traverses from vertex $(i, t)$ to vertex $(j, t+TT_f(i, j))$. For intermediate cells not allowing for dwelling (i.e. $(i, j) \notin \Omega$), they only have constant cell running time of $TT_f(i, j) = FT_f(i, j)$. For cells allowing for dwelling (typically with downstream nodes at stations), there are a set of cell running time constrained by minimum and
maximum waiting times, that is, \( w_{f}^{\text{min}}(i,j) + FT_{f}(i,j) \leq TT_{f}(i,j) \leq w_{f}^{\text{max}}(i,j) + FT_{f}(i,j) \). This type of arcs ensures constraints (26), (27), (28), (29), (30) and (31).

(2) Cell waiting arcs from \((o_{f}, t)\) to \((o_{f}, t+1)\) at the origin cell of train \(f\). The feasible time window at the node \(o_{f}\) covers from the earliest departure time \(EST_{f}\) to \(T\). By introducing the origin cell waiting arcs, we can construct a single source vertex at the origin node \(o_{f}\) and at the time instance of \(EST_{f}\). This type of arcs corresponds to constraints (24) and (25).

(3) Dummy arcs from \((s_{f}, t)\) to \((s_{f}, t+1)\) at the destination node, from the earliest departure time \(EST_{f}\) to \(T\). By introducing the dummy arcs, we can construct a single sink vertex at the time instance of \(T\).

By restricting the state transition through different types of arcs, we in fact set an infinitely large cost for those infeasible or invalid arcs so one can adapt the standard shortest path algorithm to consider additional constraints in (P2). The mapping constraints (28) between the time-space network and cell-based physical network, and the flow balance constraints (21-23) on each cell and the origin/destination nodes are automatically taken into account by the network representation. The capacity constraints (32) and (33) will be considered through the resource costs in the label correcting algorithm discussed in a later section.

For illustrative purposes, Fig. 7 depicts a simple rail network with four nodes and five cells and the corresponding extended time-space network for train \(f\). It can be seen that there are a number of possible train paths for train \(f\) starting from departure time \(t=0, 1, \ldots, 9, 10\) along traveling arcs.

Fig. 7 A simple rail network and the corresponding extended time-space network for train \(f\) from node 1 to node 4.

On the extended time-space network, main decision variables of the least cost problem can be represented by binary network flow variables \(\delta_{f}(i, j, t)\) and \(\delta'_{f}(i, j, t)\). They can be linked to cumulative flow variables by Eqs. (41) and (42).

\[
\delta_{f}(i, j, t) = a_{f}(i, j, t) - a_{f}(i, j, t - 1) \tag{41}
\]

\[
\delta'_{f}(i, j, t) = d_{f}(i, j, t) - d_{f}(i, j, t - 1) \tag{42}
\]
where \( \delta_{ij}(i,j,t)=1 \) represents train \( f \) arriving at the upstream node \( i \) of cell \( (i,j) \) at time \( t \), \( \delta_{ij}(i,j,t)=0 \), otherwise; \( \delta'_{ij}(i,j,t)=1 \) represents train \( f \) departing from the downstream node \( j \) of cell \( (i,j) \) at time \( t \). \( \delta'_{ij}(i,j,t)=0 \), otherwise.

As illustrated in Fig. 8(a), train \( f \) arrives at cell \( (2,4) \) at time 8 and departs at time 10. Through cumulative flow variables, \( a_{ij}(2,4,t)=0 \) for \( t<8 \), \( a_{ij}(2,4,t)=1 \) for \( t \geq 8 \), \( d_{ij}(2,4,t)=0 \) for \( t<10 \), and \( d_{ij}(2,4,t)=1 \) for \( t \geq 10 \). The network flow variables can then be transformed by Eqs. (41) and (42) and result in \( \delta_{ij}(2,4,t)=1 \) for \( t=8 \) and \( \delta'_{ij}(2,4,t)=1 \) for \( t=10 \), as illustrated in Fig. 8(b).

![Fig. 8 Linkage between cumulative flow variables and network flow variables](image)

5.3 Time dependent shortest path algorithm

We now present a label correcting algorithm for solving the time-dependent least cost path problem. The algorithmic framework is adapted from the algorithm by Ziliaskopoulos and Mahmassani (1993) for highway or urban road networks, where the scan eligible (SE) list only contains spatial nodes, rather than time-space vertexes. Another special feature of our proposed algorithmic framework is that we allow train waiting at cells so there are multiple possible cell running times from the same time, while the deterministic algorithm for highway networks typically considers one instance of time-dependent arc travel time. Interested readers can also refer to Pallottino and Scutellà (1998) for more details on time-space network construction and other time-dependent shortest path algorithms using different mechanisms of constructing and updating SE lists. A list of symbols is first introduced in Table 8 and the shortest path algorithm is then detailed.

**Table 8**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s )</td>
<td>origin node, corresponding to ( o_f )</td>
</tr>
<tr>
<td>( r )</td>
<td>destination node, corresponding to ( s_f )</td>
</tr>
<tr>
<td>( \Theta(i,t) )</td>
<td>the corresponding node of vertex ((i,t))</td>
</tr>
<tr>
<td>( \lambda_{ij}(j,t) )</td>
<td>the least cost (i.e. label) from vertex ((s, EST_i)) to vertex ((j,t))</td>
</tr>
<tr>
<td>( \pi_{ij}(j,t') )</td>
<td>the preceding least cost vertex of vertex ((j,t')), denoted as time-space vertex ((i,t))</td>
</tr>
<tr>
<td>( \sigma_{ij} )</td>
<td>free-flow running time of cell ((i,j)), corresponding to ( FT_{ij}(i,j) )</td>
</tr>
<tr>
<td>( \Delta_{ij}(t) )</td>
<td>waiting time of cell ((i,j)) at time ( t )</td>
</tr>
</tbody>
</table>
\[ \partial_{i,j}(t, t+\sigma_{i,j} + \Delta_{i,j}(t)) = \varrho_{i,j}^{\pm} \]
\[ \Gamma(i,t) \quad \text{set of outgoing vertexes of vertex } (i, t) \]

**Input:** Networks $G$ and TSG, origin node $s$ (i.e., $s_f$), destination node $r$ (i.e., $s_f$), starting time $\text{EST}_f$, and resource cost vector $\varrho$ at current iteration.

**Output:** The least cost path from $s$ to $r$ at time $t$.

**Step 1. (Initialization)**

Create an empty SE list; Set $\lambda_i(j,t) = \infty$, $\forall j \in N \setminus \{s\}, t=1,\ldots,T$; $\lambda_i(s,t) = 0$, $\forall t = 1,\ldots,T$; $\pi_i(s,t) = \emptyset$, $\forall t = 1,\ldots,T$; Insert the source vertex $(s, t)$ into the SE list.

**Step 2. (Label updating)**

While SE list is not empty do

Pop up the front vertex from the SE list, denoted by $(i, t)$.

For vertex $(j,t) \in \Gamma(i,t)$, do

For $t = \text{EST}_f$ to $T$

If $\Theta(j,t+\sigma_{i,j} + \Delta_{i,j}(t)) = r$ Then

Set candidate new cost label by

\[ \lambda'_i(j,t+\sigma_{i,j} + \Delta_{i,j}(t)) = \lambda_i(i,t) + \partial_{i,j}(t, t+\sigma_{i,j} + \Delta_{i,j}(t)) + |t+\sigma_{i,j} + \Delta_{i,j}(t) - D(s) - D(s)| \]

Else

Set candidate new cost label by

\[ \lambda'_i(j,t+\sigma_{i,j} + \Delta_{i,j}(t)) = \lambda_i(i,t) + \partial_{i,j}(t, t+\sigma_{i,j} + \Delta_{i,j}(t)) \]

End

If $\lambda'_i(j,t+\sigma_{i,j} + \Delta_{i,j}(t)) < \lambda_i(j,t+\sigma_{i,j} + \Delta_{i,j}(t))$ Then

Set node cost label by

\[ \lambda_i(j,t+\sigma_{i,j} + \Delta_{i,j}(t)) = \lambda'_i(j,t+\sigma_{i,j} + \Delta_{i,j}(t)) \]

Update preceding vertex by setting $\pi_i(j,t+\sigma_{i,j} + \Delta_{i,j}(t))$ to time-space vertex $(i, t)$;

If vertex $(j, t')$, i.e., vertex $(j,t+\sigma_{i,j} + \Delta_{i,j}(t))$, has been in the SE list,

Then

Add vertex $(j, t')$ to the front of SE list;

Else

Add vertex $(j, t')$ to the back of SE list;

End

End // Updating node cost label

End // for each cell waiting time

End // for each possible starting time

End // for each vertex

Remove vertex $i$ from the SE list.

End
Step 3. (Fetch the time dependent shortest path)

**Step 3.1** Find the vertex \((j^*, t^*)\) corresponding to the destination node \(r\) and with the least cost; Set vertex \((j^*, t^*)\) as the current vertex \((k, t)\);

**Step 3.2** Backtrack from the destination node \(r\) to node \(s\);

While vertex \((k, t)\) is not corresponding to the origin node \(s\)

1. Find the preceding vertex \((i, t')\) of the current vertex \((k, t)\);
2. Update preceding vertex \((i, t')\) as the current vertex \((k, t)\).

End

**Step 3.3** Reverse the backward path and output the least cost path from \(s\) to \(r\) at \(t\);

**Step 3.4** Terminate the algorithm.

For illustrative purpose, Fig. 9 depicts the process of fetching the least cost path on the physical network (illustrated in Fig. 7) for train \(f\). Train \(f\) departs from its origin node 1 with the earliest departure time 0, and aims to arrive at its destination node 4 at the scheduled time 2. Each physical node is extended to a sequence of vertexes. For example node 4 is extended to vertexes \((4,0), (4,1), (4,2), (4,3), (4,4)\) and \((4,5)\). All resource prices (i.e., Lagrangian multipliers) are assumed to be 0.

After the label correcting process in step 2, each vertex has its least cost label and preceding vertex. For the destination node 4, the vertex \((4,4)\) holds the least cost of 2 and the preceding vertex \((3,3)\), as shown in bold in Fig. 9. Following Step 3 in the above procedure, one can further find the preceding vertex of \((3,3)\) as \((1,0)\). Since vertex \((1,0)\) corresponds to the origin node 1, the algorithm is terminated. Then one can easily reverse the backward and output the time-space least cost path as node sequence 1<->3<->4 with timestamps of minutes 0<->3<->4.

Fig. 9 Fetching the least cost time-space path from node 1 to 4 for the physical network in Fig. 7, for train \(f\) with the earliest departure time at min 0.

### 5.4 Priority rule-based implementing algorithm

At each Lagrangian iteration, we also construct a feasible solution based on priority rules to improve the upper bound estimate of the optimal solution. The algorithm is detailed as below.

**Input:** Network \(G\), train set \(F\), origin node \(o_f\), destination node \(s_f\), earliest departure time \(EST_f\) for each train \(f\).
**Output:** The routes and passing times at each station for each train \( f \), and the updated upper bound.

**Step 1. (Train priority ranking)**

Rank the trains by decreasing values of Lagrangian profits. The Lagrangian profit of each train is the ratio of total free-flow travel time divided by total travel time in the dual solution.

**Step 2. (Schedule trains one by one)**

**Step 2.1** For the train \( f^* \) with the highest priority, apply the shortest path algorithm introduced in section 5.3 to find its route and passing times at each station;

**Step 2.2** Fix the route and passing times at each station for \( f^* \); Record the capacity usage of \( f^* \) on network \( G \);

**Step 2.3** If all trains have been scheduled, move to Step 3, otherwise, loop back to Step 2.1.

**Step 3. (Update and output upper bound)**

**Step 3.1** Compute the objective value of the heuristic solution obtained by step 2;

**Step 3.2** Update the upper bound using the new objective value;

**Step 3.3** Output the route, passing times at each station, and the new upper bound at the current Lagrangian iteration.

**6. Numerical experiments**

**6.1 Experimental setup**

All the constraints in problem (P1) and (P2) are implemented as integer programming models through a commercial solver CPLEX with version number 12.3. The proposed Lagrangian relaxation solution framework, which includes a shortest path algorithm and priority rules, is implemented as a customized algorithm by Visual C++ 2008 on a Windows 7 X64 professional platform and integrated into a software package named FastTrain. Both source codes can be found at NTM (2013) and FastTrain (2013).

We first adapt a network (as shown in Fig. 10) from the INFORMS RAS problem competition in 2012 (INFORMS RAS, 2012) as one of the test datasets. It consists of 76 nodes and 85 cells, with a total track length of 137.6 km. MOW in Fig. 10 represents Maintenance of Way where the corresponding cells are unavailable to train traffic due to repair or inspection activities. It should be remarked, the original problem statement provided by INFORMS RAS (2012) uses a more realistic but complex non-linear objective function that considers different penalties for early vs. late schedule delays. In this paper, we use a relatively simple linear objective function in (P1) and (P2), i.e. minimize total deviation time for all trains in the following experiments.
In order to further evaluate the comparative benefits of the REF-SRR model, we construct a more complex rail network as illustrated in Fig. 11. It consists of 85 nodes and 97 cells, with a total track length of 287.7 km and one more MOW incident at cell (87, 90). For more detailed information about the networks in Fig. 10 and Fig. 11, we refer to the data set released in FastTrain (2013).

The initial delays of trains occur at origin stations and are generated according to the uniform distribution with a range between 5 and 10 min. The number of random instances varies from 5 to 10. For simplicity, we set the safety headway \( g=0 \) min and \( h=3 \) min to be consistent with the input data description by INFORMS RAS, 2012, although \( g \) can be set a different value to fully represent the time interval between the blockage of a block section and the actual arrival at the block section.

All the following experiments are performed on a Lenovo ThinkPad X230 laptop with 2.9 GHz Intel i7 CPU and 4 GB memory.

### 6.2 Efficiency and effectiveness analysis of Lagrangian solution framework

We are interested in solution quality, computational time and model complexity for the following four different types of solution methods.

1. IP-CPLEX(M): integer programming (IP) implementation of (P1: big-\( M \) method), solved by CPLEX;
2. IP-CPLEX: integer programming implementation of (P2: cumulative flow variable-based model), solved by CPLEX;
3. LP-CPLEX(M), linear programming (LP) relaxation of (P1: big-\( M \) method), solved by CPLEX;

Fig. 10 An experimental rail network adapted from INFORMS RAS (2012).

Fig. 11 A more complex experimental rail network.
(4) LR-C++, integer programming implementation of (P2: cumulative flow variable-based model) and solved by our customized C++ package with a built-in shortest path algorithm in a Lagrangian relaxation (LR) solution framework.

(1) Solution quality and computational time of IP-CPLEX(M), LP-CPLEX(M) and LR-C++

We use the network in Fig. 10 as the test bed and consider different numbers (from 2 to 20) of trains and planning horizon \( T = 1440 \) min.

The lower bound measure is examined in Table 9, and both LP-CPLEX(M) and LR-C++ can obtain lower bounds within 2.5 minutes. LR-C++ can further improve lower bound quality with an average of 47.12\% compared to LP-CPLEX(M), with consuming a little more computational time (up to 1 minute).

Table 9
Lower bound (in minute) of LP-CPLEX(M) and LR-C++.

<table>
<thead>
<tr>
<th>Number of trains</th>
<th>LP-CPLEX(M)</th>
<th>LR-C++</th>
<th>(LR-C++ - LP-CPLEX(M)) / LP-CPLEX(M)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>28</td>
<td>28</td>
<td>0.00%</td>
</tr>
<tr>
<td>4</td>
<td>47</td>
<td>60.58</td>
<td>28.89%</td>
</tr>
<tr>
<td>6</td>
<td>75</td>
<td>90.63</td>
<td>20.84%</td>
</tr>
<tr>
<td>8</td>
<td>90</td>
<td>111.17</td>
<td>23.52%</td>
</tr>
<tr>
<td>10</td>
<td>95</td>
<td>123.60</td>
<td>30.11%</td>
</tr>
<tr>
<td>12</td>
<td>112</td>
<td>136.69</td>
<td>22.04%</td>
</tr>
<tr>
<td>14</td>
<td>92</td>
<td>227.76</td>
<td>147.57%</td>
</tr>
<tr>
<td>16</td>
<td>95</td>
<td>232.13</td>
<td>144.35%</td>
</tr>
<tr>
<td>18</td>
<td>130</td>
<td>163.54</td>
<td>25.80%</td>
</tr>
<tr>
<td>20</td>
<td>128</td>
<td>163.96</td>
<td>28.09%</td>
</tr>
</tbody>
</table>

The overall quality of upper bounds is shown in Table 10. First, IP-CPLEX(M) can find optimal solutions for smaller cases (e.g. number of trains less than 10). We also observe that, when the number of trains is larger than 10, IP-CPLEX(M) cannot obtain solutions after 3 hours (please see Table 11). This demonstrates the limited capability of IP-CPLEX(M) on finding solutions for a relatively large rail network. In comparison, LR-C++ can find feasible solutions with very reasonable gaps (up to 9.59\%) for all the test cases (if the optimal solution is available) within about 1.3 minutes. The excellent capability of LR-C++ on finding feasible solutions is further confirmed by larger-network experiments (with up to 40 trains) shown in section 6.3.

Table 10
Upper bound (in minute) of IP-CPLEX(M) and LR-C++.

<table>
<thead>
<tr>
<th>Number of trains</th>
<th>IP-CPLEX(M) (Optimal)</th>
<th>LR-C++ (feasible)</th>
<th>Gap of feasible solution to optimal solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>28</td>
<td>28</td>
<td>0.00%</td>
</tr>
<tr>
<td>4</td>
<td>68</td>
<td>71</td>
<td>4.41%</td>
</tr>
<tr>
<td>6</td>
<td>96</td>
<td>104</td>
<td>8.33%</td>
</tr>
<tr>
<td>8</td>
<td>131</td>
<td>137</td>
<td>4.58%</td>
</tr>
<tr>
<td>10</td>
<td>146</td>
<td>160</td>
<td>9.59%</td>
</tr>
<tr>
<td>12</td>
<td>-</td>
<td>241</td>
<td>-</td>
</tr>
<tr>
<td>14</td>
<td>-</td>
<td>361</td>
<td>-</td>
</tr>
<tr>
<td>16</td>
<td>-</td>
<td>413</td>
<td>-</td>
</tr>
<tr>
<td>18</td>
<td>-</td>
<td>289</td>
<td>-</td>
</tr>
<tr>
<td>20</td>
<td>-</td>
<td>395</td>
<td>-</td>
</tr>
</tbody>
</table>

“-” means no optimal solution obtained by IP-CPLEX(M) within 10800 seconds (3 hours) and accordingly the corresponding optimality gap measure is not available.
Table 1
Computational time (in second) for lower bound and upper bound.

<table>
<thead>
<tr>
<th>Number of trains</th>
<th>Computational time for lower bound</th>
<th>Computational time for upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LP-CPLEX(M)</td>
<td>LR-C++</td>
</tr>
<tr>
<td>2</td>
<td>4.37</td>
<td>&lt;1</td>
</tr>
<tr>
<td>4</td>
<td>5.39</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>7.68</td>
<td>23</td>
</tr>
<tr>
<td>8</td>
<td>9.67</td>
<td>46</td>
</tr>
<tr>
<td>10</td>
<td>14.50</td>
<td>73</td>
</tr>
<tr>
<td>12</td>
<td>20.22</td>
<td>80</td>
</tr>
<tr>
<td>14</td>
<td>27.29</td>
<td>97</td>
</tr>
<tr>
<td>16</td>
<td>40.14</td>
<td>100</td>
</tr>
<tr>
<td>18</td>
<td>60.51</td>
<td>120</td>
</tr>
<tr>
<td>20</td>
<td>91.44</td>
<td>140</td>
</tr>
</tbody>
</table>

"-" means no solution obtained within 10800 seconds (3 hours).

(2) Complexity analysis of IP-CPLEX(M) and IP-CPLEX

We next analyze the complexity of integer programming implementation of (P1) and (P2), in terms of the number of constraints and variables. The small network illustrated in Fig. 1 is used as the test bed and we consider a number (up to 10) of trains and planning horizon \( T = 300 \) min.

As listed in Table 12, the number of constraints, variables, and computational time increase for both models with an increase in the number of trains. It can be further seen that the number of constraints, variables, and computational time for IP-CPLEX(M) is much less than IP-CPLEX. This demonstrates the advantage of the big-M method compared to the cumulative flow variable-based formulation on a small rail network, although IP-CPLEX(M) has limited capability in solving the dispatching problem on a large network with a large number of trains, as demonstrated in Tables 9, 10 and 11. This implies that one can directly use IP-CPLEX(M) to solve dispatching problems with a small number of if-then type decision variables for the safety headway requirements, e.g.: (1) a small number of trains on a relatively large network, for example, 6 trains on the network in Fig. 10, or (2) a large number of trains on a small network, for instance, 10 trains (or more) on the network in Fig. 1.

Table 12
Number of constraints, variables and computational time for IP-CPLEX(M) and IP-CPLEX.

<table>
<thead>
<tr>
<th># of trains</th>
<th>IP-CPLEX(M)</th>
<th>IP-CPLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># of variables</td>
<td># of constraints</td>
</tr>
<tr>
<td>2</td>
<td>161</td>
<td>350</td>
</tr>
<tr>
<td>4</td>
<td>453</td>
<td>1052</td>
</tr>
<tr>
<td>6</td>
<td>877</td>
<td>2106</td>
</tr>
<tr>
<td>8</td>
<td>1433</td>
<td>3512</td>
</tr>
<tr>
<td>10</td>
<td>2121</td>
<td>52700</td>
</tr>
</tbody>
</table>

"-" means no solution obtained within 10800 seconds (3 hours).

It is interesting to quantify how the size of the proposed (P2), i.e., REF-SRR model, depends on the length of the planning horizon as the cumulative flow variable-based representation dramatically extends the number of variables along the time dimension. We examine the impact of planning time horizon \( T \) on the number of constraints and variables of the REF-SRR model by taking the test case of 2 trains on the small network in Fig. 1 as an example. As illustrated in Fig. 12, when the planning time horizon becomes longer, the number of constraints and variables increases in a linear manner. This
implies that one can speed up the solution process by choosing an appropriate planning time horizon that should only cover the entire trips of all involved trains.

Fig. 12 Number of constraints and variables of the REF-SRR model corresponding to different T (2 trains, the network in Fig. 1).

(3) Identification of the dominating hard constraints

The next focus is on computational time and solution quality associated with different relaxation techniques. The small network illustrated in Fig. 1 is used as the test bed and 4 trains need to be dispatched within a planning horizon $T = 300$ minutes. The experimental results are listed in Table 13. First, we use CPLEX to solve the REF-SRR model with exact binary variable constraints, which takes about 61.66 seconds. We are then interested in examining the strength of three relaxation methods: (1) relaxing the cell capacity constraints that couple different trains on the same cell (2) relaxing the cell-to-cell transition constraints that enforce the dynamic flow balance/connection relationship, and (3) relaxing binary decision variables constraints. The quality of lower bound is measured by the percentage gap between a lower bound estimate $Z^{LB}$ and the corresponding optimal value $Z^*$. All the relaxations (Models B–D) need about 19.33% to 44.45% of the computational time for the original IP model (Model A), but provide lower bounds with quality reduction from 51.85% to 100%. Among the three types of constraints relaxation, the cell capacity constraints seem to be dominating “hard” constraints, as Model B (with the capacity constraints being relaxed) uses the least time to solve and provides the tightest lower bound estimator (48.15%).

Table 13
Computational time and solution quality for different relaxation types (4 trains with $T = 300$ min).

<table>
<thead>
<tr>
<th>Model</th>
<th>Computational time, unit: second (percentage of time compared to original IP REF-SRR formulation)</th>
<th>Solution quality, unit: minute (percentage of lower bound $Z^{LB}$ / optimal value $Z^*$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Original IP REF-SRR formulation</td>
<td>61.66 (100%)</td>
<td>27 (100%)</td>
</tr>
<tr>
<td>B. IP with relaxation of cell capacity constraints (33)</td>
<td>11.92 (19.33%)</td>
<td>13 (48.15%)</td>
</tr>
<tr>
<td>C. IP with relaxation of cell-to-cell transition</td>
<td>16.20 (26.27%)</td>
<td>0 (0%)</td>
</tr>
</tbody>
</table>
6.3 Benefits of simultaneous train rerouting and rescheduling

This section aims to demonstrate the comparative benefits of the REF-SRR model on an $N$-track network. Recall that the REF-SRR model can equivalently consider all possible routes on the network for problem (P2), while the traditional sequential rerouting and rescheduling approaches compute schedules under specifying a number of routes $K$ for each train $f$, where $K=|P_f|$. Note that $P_f$ contains a number of routes sorted in ascent order according to route length. For instance, $K=1$ means that there is one route (i.e., the shortest one) from origin to destination in $P_f$. Obviously, the REF-SRR model offers solutions that are at least better than or the same as those of SEQ model, as the solution space of the REF-SRR model contains the sequential approach’s space as a subset.

We first use the relatively complex network illustrated in Fig. 10 to numerically evaluate the comparative benefits of two methods in the cases of $K=1, 3, 5$ and 10 for the SEQ model. In a 24-hour period, 20 trains need to be dispatched. In the follow set of experiments, the initial delays of trains follow the uniform distribution with a range between 10 and 20 minutes.

Fig. 13 and Fig. 14 examine the upper bound and optimality gap, respectively, under different model settings. In general, we should expect better solution quality with more route options. First, we consider the SEQ model with different route size $K$. Compared to $K=1$, with increasing $K$ for each train, the upper bound solution quality improves 6.8%, 26.4% and 32.1% corresponding to $K=3, 5$, and 10. Second, we compare the SEQ model with $K=10$ with the REF-SRR model that all possible available routes. Interestingly, one can still observe additional benefits of about an improved upper bound solution by 10.9% and a 7.4% gap reduction. This result indicates that it is extremely valuable to consider and use the REF-SRR model as a solution benchmark for evaluating existing solution strategies typically obtained by a sequential scheduling process or relatively limited available route sets.

Furthermore, in order to evaluate the quality of the feasible solutions obtained by the priority rules-based algorithm, we develop another heuristic algorithm using First-In-First-Out (FIFO) rules. The objective value obtained by the proposed rules-based algorithm is $(3381-2972)/3381*100\% = 12.10\%$ better than the objective values obtained by FIFO rules.
Fig. 13 Upper bound obtained by means of different model settings (with 20 trains).

Fig. 14 Optimality gap obtained by means of different model settings (with 20 trains).

Fig. 15 Upper bound and lower bound with computational time (with all possible paths, 20trains).
Moreover, thanks to the efficient implementation of the proposed LR-C++ implementation, as illustrated in Fig. 15, the simultaneous rerouting and rescheduling method can find a good upper bound quickly within less than 100 iterations, which takes less than 1 minute of CPU time. Moreover, the lower bound also quickly becomes stable within 5 minutes, which gives optimality information of the upper bound.

We next use the more complex network in Fig. 11 to further examine the comparative benefits of the REF-SRR model. In this instance, 40 trains need to be dispatched within a planning horizon of 24 hours. The initial delays of trains follow a uniform distribution with a range between 10 and 20 minutes.

As illustrated in Fig. 16, for a more complex rail network and a larger number of trains, the simultaneous rerouting and rescheduling method can find an upper bound (feasible solution) at 5 minutes with an optimality gap 33.90%. We can also see that the lower bound and upper bound tend to become better with an increase of computational time (the number of Lagrangian iterations), which means that if more powerful computing resources are available, a smaller optimality gap may be obtained within 5 minutes, which is a generally acceptable computational time for the real-time train dispatching problem. Moreover, the objective value obtained by the proposed rules-based algorithm is (5259-3897)/5259*100% = 25.90% better than the objective values obtained by FIFO rules.

Fig. 16 Upper bound and lower bound with computational time (with all possible paths, 40 trains).

It’s interesting to investigate why the optimality gap is relatively large for a large network and a large number of trains. There are two possible reasons: (1) the lower bound is optimized by the subgradient method which is proved to be weaker than an enhanced lower bound searching method (Zhou and Zhong, 2007). However, extending the enhanced method from a single-track line into the context of an N-track network is quite complicated with an additional dimension of track re-routing, and (2) the upper bound is obtained by a priority rules-based algorithm rather than by a rigorous branch and bound algorithm. According to our experiments which use the latter function, the optimality gap is dramatically reduced to 21% in our experiments, compared to the one using the total deviation objective function.
7. Concluding remarks

This paper is motivated by the modeling need for dispatching trains on a general N-track rail network. We proposed two innovative mathematical programming models for simultaneous train rerouting and rescheduling, one based on the standard big-M method, the other based on network-wide cumulative flow variables. We then presented a Lagrangian relaxation solution algorithm that embeds a computationally efficient time-dependent shortest path algorithm for solving the path-finding subproblems for each train in a time-space network. The benefits of simultaneous train rerouting and rescheduling model are assessed in comparison with commonly used sequential scheduling approaches. The experimental results show that the proposed formulations perform well for dispatching trains on a general N-track rail network.

The recent paper by Mu and Dessouky (2011), mentioned in the introduction, is closely related to our proposed methodology. The differences between their “FixedPath” and “Flexible” models and our proposed models can be summarized as below:

1. The “FixedPath” and “FlexiblePath” models are conflict-oriented, in which “if-then” constraints are used to represent train orders and ensure safety headway constraints. In our proposed REF-SRR model, cumulative flow variables are introduced to represent the usage of time-space resources. More precisely, a certain set of resources can be only used by a time-space path of a train, which enforces the minimum headway constraints in a time-space network. By avoiding using a large number of pair-wise train order (binary) variables, the REF-SRR model enables an efficient and relatively easy problem decomposition mechanism. In each train-specific subproblem, we only need to handle a smaller variable space and it is also possible to utilize the shortest path algorithms to find relaxed solutions quickly.

2. To model train order constrains in network settings, Mu and Dessouky (2011) used specific assumptions on the directions of tracks (e.g. West vs. East). In their paper, to further simplify the path selection mechanism for the “FlexiblePath” model, each track can only allow trains travelling in one direction. Without the above two assumptions, the proposed SRR and REF-SRR models allow any feasible paths on a complex network with both unidirectional double tracks and bidirectional single tracks.

3. Both “FixedPath” and SEQ models adopt a two-stage scheduling process: determining the train path set, and then computing the detailed train paths. However, the SEQ model allows a train to select a path from a restricted path set with multiple paths (in order to incorporate dispatching preferences), while the “FixedPath” model assigns a train to exactly one path.

Our future research will address the following main extensions.

1. One can make the representation of rail infrastructure more realistic, e.g., 1) respect the relations between track circuits imposed by signaling systems 2) consider sectional release of a route, and 3) take into account train length, acceleration and deceleration times.

2. Based on the proposed mathematical programming models, different optimization or reformulation methods can be further utilized to improve the solution quality and computational efficiency. In particular, the proposed priority-rule based heuristic algorithm and Lagrangian relaxation estimation methods should be further improved to obtain better feasible solutions (i.e., upper bound). Stronger lower bound estimators are critically needed to reduce the relatively large optimality gaps that still exist for some instances in this paper.

3. As studied in Meng and Zhou (2011), it’s challenging to provide practically useful strategies for managing network train traffic with consideration of the dynamicity and stochasticity of
perturbation information. Scenario-based rolling horizon solution approaches can be further developed to solve the train dispatching problem on a general rail network under a dynamic and stochastic environment.

(4) In this paper, the granularity of time is one minute. Considering shorter time intervals (e.g., 15 seconds) in a time-expanded network lead to a much larger variable space, which could further dramatically increase computational time. To meet real-time dispatching needs, one promising research direction is how to incorporate parallel computing techniques into the proposed solution algorithms so as to speed up the solution process, where multiple subproblems in the relaxed problem can be easily handled by individual CPU cores in parallel.

Acknowledgements

The work of the first author was jointly supported by National Natural Science Foundation of China (No. 71201009), State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University (No. RCS2012ZT003, RCS2013ZZ001), Beijing Higher Education Young Elite Teacher Project (YETP0581) and Research Fund for the Doctoral Program of Higher Education of China (No. 20120009120015). The authors would like to thank three anonymous referees and Associate Editor Dr. Malachy Carey for their constructive suggestions, which improve the content of this paper substantially. The generous assistance of our colleague Ms. Xiaojie Luan is greatly appreciated. The authors are of course responsible for all results and opinions expressed in this paper.

Appendix

A sequential train rerouting and rescheduling model based on network cumulative flow variables

The decision variables in Table 14 are used specifically for sequential train rerouting and rescheduling with a given route set \( P_f \) for each train \( f \).

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y_f^p(m,t) )</td>
<td>0-1 binary time-space occupancy variables along a route, ( y_f^p(m,t) = 1 ), if train ( f ) occupies the ( m )-th cell along route ( p ) at time ( t ), and otherwise ( y_f^p(m,t) = 0 )</td>
</tr>
<tr>
<td>( a_f^p(m,t) )</td>
<td>0-1 binary cumulative arrival flow variables along a route, ( a_f^p(m,t) = 1 ), if train ( f ) has already arrived at the ( m )-th cell along route ( p ) by time ( t ), and otherwise ( a_f^p(m,t) = 0 )</td>
</tr>
<tr>
<td>( d_f^p(m,t) )</td>
<td>0-1 binary cumulative departure flow variables along a route, ( d_f^p(m,t) = 1 ), if train ( f ) has already departed from the ( m )-th cell along route ( p ) by time ( t ), and otherwise ( d_f^p(m,t) = 0 )</td>
</tr>
</tbody>
</table>

While the simultaneous train rerouting and rescheduling models proposed in sections 3 and 4 do not impose a pre-specified route set constraint, the sequential train rerouting and rescheduling model to be detailed here involves a two-stage process. First, one can determine a route set \( P_f \) (with a limited number of routes) for each train \( f \) based on previous experiences from dispatchers or other practical rules. Each route \( p \) has a sequence of \( N_f^p \) cells with cell sequence number \( m = 1, \ldots, N_f^p \). With this pre-specified route set \( P_f \), the second stage of the sequential train rerouting and rescheduling model aims to select a route \( p \) from set \( P_f \) and also jointly determine arrival/departure times along the route.
Some related studies on network-wide train scheduling can be viewed as a special case of the sequential train rerouting and rescheduling model, where route set $P_f$ only contains a single route for each train but the route can be updated iteratively to approach the final close-to-optimal solutions.

One may consider to include all possible routes in a network in $P_f$, to transform the sequential rerouting/rescheduling model to the simultaneous rerouting/rescheduling model. Unfortunately, it's very difficult to explicitly enumerate all possible routes in a network and achieve the transformation, since the number of routes in a network significantly increases with the number of stations and segments.

By introducing the binary variables $a_f^p(m,t)$, $d_f^p(m,t)$ and $y_f^p(m,t)$ for each cell sequence number $m$ along route $p$, the sequential train rerouting and rescheduling model can be reformulated as (P3).

$$Z = \min \sum_{f} \sum_{p} \sum_{t} \left[ t \times [d_f^p(N_p,t) - d_f^p(N_p,t-1)] - D(s_f) \right]$$

Subject to

**Group I: Flow balance constraints**

Route selection constraints at the origin node:

$$\sum_{p = f} a_f^p(1,t) = 1, \forall f, t \geq EST_f$$

**Group II: Time-space network constraints**

Starting time constraints at the origin node:

$$a_f^p(1,t) = 0, \forall f, p, t < EST_f$$

$$d_f^p(1,t) = 0, \forall f, p, t < EST_f$$

Within cell transition constraints:

$$d_f^p(m,t + FT_f(\phi_f^{p,m})) \leq a_f^p(m,t), \forall f, p, m, t$$

Within route cell-to-cell transition constraints:

$$a_f^p(m+1,t) = d_f^p(m,t), \forall f, p, m + 1 \leq N_p, t$$

**Group III: Running time and dwell time constraints**

Running time constraints:

$$TT_f(\phi_f^{p,m}) = \sum_{t} \left[ t \times [d_f^p(m,t) - d_f^p(m,t-1)] \right] - \sum_{t} [t \times [a_f^p(m,t) - a_f^p(m,t-1)]], \forall f, p, m$$

Minimum running time constraints:

$$TT_f(\phi_f^{p,m}) \geq FT_f(\phi_f^{p,m}), \forall f, p, m$$

Minimum and maximum dwell time constraints:

$$w_f^{\min}(\phi_f^{p,m}) + FT_f(\phi_f^{p,m}) \leq TT_f(\phi_f^{p,m}) \leq w_f^{\max}(\phi_f^{p,m}) + FT_f(\phi_f^{p,m}), \forall f, \phi_f^{p,m} \in \Omega$$

**Group IV: Capacity constraints**

Cell occupancy indication constraints:

$$y_f^p(m,t) = a_f^p(m,t + g) - d_f^p(m,t - h), \forall f, p, m, t$$

Cell capacity constraints:

$$\sum_{f' = f', p, m} y_f^p(m',t) \leq Cap(\phi_f^{p,m}, t), \forall f, p, m, t$$

**Group V: Time-connectivity constraints for cumulative flow variables**

$$a_f^p(m,t) \geq a_f^p(m,t-1), \forall f, p, m, t$$
\[ d_f^p(m,t) \geq d_f^p(m,t-1), \forall f, p, m, t \] (55)

Similar to variables \( a_f(i,j,t+g) \) and \( d_f(i,j,t-h) \), the variables \( d_f^p(m,t+g) \) and \( d_f^p(m,t-h) \) represent the occupancy of train \( f \) on the \( m \)-th cell along route \( p \).

Obviously, the flow balance constraints (21)-(23) and (26)-(27) are critical for (P2), and the flow conservation relationship is guaranteed in (P3) through Eqs. (44), (47) and (48) along route \( p \). In particular, Eq. (44) can ensure that only one route is selected from the pre-specified set of routes at the origin node of each train.

The constraints (45)-(55) for (P3) on the time-space network, running time, dwell time, cell capacity and time-connectivity of cumulative flow variables are similar to the corresponding constraints (24)-(27) and (29)-(35) for (P2).

If one sets \( |p_f|=1 \), i.e., only one exact route \( p \) is specified for train \( f \), the above sequential model becomes a traditional sequential model in which there is no choice of routes and no route subscripts.

References


