Transportation Research Part C 70 (2016) 197–213

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Modeling strategies for effectively routing freight trains through complex networks

Pavankumar Murali [a], Fernando Ordóñez [b,*], Maged M. Dessouky [c]

[a] *IBM T. J. Watson Research Center, 1101 Kitchawan Road, Rt 134, Yorktown Heights, NY 10598, United States*
[b] *Industrial Engineering Department, Universidad de Chile, República 701, Santiago 8370439, Chile*
[c] *Department of Industrial & Systems Engineering, University of Southern California, 3715 McClintock Avenue, Los Angeles, CA 90089, United States*

A R T I C L E   I N F O

Article history:
Received 30 April 2014
Received in revised form 29 July 2015
Accepted 31 August 2015
Available online 2 October 2015

Keywords:
Train operations
Routing
Integer model
Heuristics

A B S T R A C T

An important factor in an efficient operation of freight railroad companies is their ability to obtain routes and schedules that improve rail network capacity utilization. In this paper we present a decision tool to aid train planners obtain quickly good quality routes and schedules for short time horizons (e.g., daily) to better manage the limited track capacity available for train movements. This decision tool is made up of an integer programming (IP)-based capacity management model and a genetic algorithm (GA)-based solution procedure. The capacity management model assigns trains to routes based on the statistical expectation of running times in order to balance the railroad traffic. The GA procedure determines the best initial routes and release times for trains to depart from origin stations and enter a network, given travel time estimates across aggregated sections of a network. We test our modeling technique by comparing the travel times obtained for a network in Los Angeles using these initial routes and release times, with those obtained from a simulation model, presented by Lu et al. (2004), which has been shown to be representative of the real-world travel times. Our experimental results show that our recommended solution procedure is capable of lowering travel times, as estimated by Lu et al. (2004), by up to 20%.

1. Introduction

According to a study conducted by Hillestad et al. (2009), railroads contribute to transporting approximately 40% of the freight traffic in the United States, by freight ton-miles. The American Association of Railroads (AAR) expects the freight rail volume to dramatically increase by 2035, resulting in 55% of the rail network being congested. To effectively manage the capacity of the rail network system it is important to consider the problem of routing and scheduling trains at the macro level taking into account multiple rail routes for a given origin and destination. For example, in Southern California there are three distinct rail lines from the Colton crossing area to the Downtown area (two served by Union Pacific and one by BNSF). The capability to balance the freight rail traffic along the three routes has the potential to significantly reduce train travel times in the area.

Our research falls in the broad area of routing and scheduling trains as part of the railway *planning* process. The focus of this paper is the development of an optimization-based approach that can aid train planners to quickly develop good quality routes and schedules over medium-scale networks, which are 30–50 miles long, to better manage the limited track capacity

* Corresponding author.
E-mail addresses: pavanm@us.ibm.com (P. Murali), fordon@dii.uchile.cl (F. Ordóñez), maged@usc.edu (M.M. Dessouky).

http://dx.doi.org/10.1016/j.trc.2015.08.022
0968-090X/© 2015 Elsevier Ltd. All rights reserved.

available for train movements. Given the daily traffic volume, train mix or heterogeneity, and sets of origins and destinations, the integrated routing and capacity management model can be solved to obtain the routes the trains should travel on and their order of departure from the origin stations. This problem is akin to a *no-wait job-shop scheduling problem with alternative routes*, where the trains are similar to jobs that need to be processed and the rail-tracks are resources such as machines, with each job J_i having a processing time p_{ik} on machine M_k. The primary difference is the scale of the problem. That is it is not computationally feasible to model each track segment as a single resource (e.g., machine). Therefore, we aggregate multiple track segments into a single resource. In a typical job-shop scheduling problem, the order of release of the jobs impacts the overall delay in processing all of them. Hence, determining the release times of our trains from their respective origins is vital to minimize travel-times. Similar to job-shop scheduling, trains are scheduled to depart from origin and intermediate stations such that the travel-times in the network are minimized. It could even be necessary to hold back a train ready to depart and instead allow another train to depart ahead, because doing so might reduce interferences between trains somewhere down in the network. However, once a train has been released into a network, a simple greedy scheduling algorithm that moves a train from one track segment to the next tends to work well in practice since holding a train reduces track capacity. Of course, there may be number of valid reasons to hold a train at intermediate track segments (e.g., to allow a higher priority passenger train pass), but these types of decisions are typically made in real-time and by dispatchers, and is outside the scope of this paper.

Fig. 1 is a section of a medium-scale network from the Los Angeles-area rail network. The network is between downtown Los Angeles and Colton crossing. There are three main routes in this section of the network – UP-Alhambra, UP-San Gabriel and BNSF. The first two are operated by Union Pacific (UP) and the third by Burlington Northern Santa Fe (BNSF), the two largest railroad companies in the United States. Each of the three routes is approximately 60 miles long. UP trains can run on the segment owned by BNSF due to an existing agreement between the two. We note that the network shown in Fig. 1 is an example network; a single railroad company can also own multiple lines. The *routing* decision determines the train track route that minimizes expected travel time given a set of possible routes going from an origin to a destination or a set of destinations. Delays occur when trains traveling in the train network interfere with each other, thus requiring one of the trains to stop or be pulled over for the other to cross or overtake it. The time taken to decelerate or stop, the duration for which the train is stationary, and the time taken to accelerate together constitute delay. We consider routing at a macro level, that is, it does not deal with which siding to travel on, which crossing should be used, which track segment of a double- or triple-track the train should travel on, etc. As an example, the routing problem for the network shown in Fig. 1 would involve determining the optimal route (Alhambra line, San Gabriel line or BNSF line) for each train departing from Colton and traveling to the Alameda Corridor. It is important to note that an underlying problem to be solved is balancing trains along the three routes depending on the expected travel times. This is very closely associated with determining the optimal release or departure times from the origin station, Colton. Releasing trains too close to each other would drastically increase travel times. To tackle this issue, we solve a joint routing-scheduling problem. As a whole, this research represents an effort in developing a quantitative model to tactically plan the movement of trains through a complex network, with decisions based on an accurate representation of the delays these trains cause on the railroad. Mu and Dessouky (2011) develop a model that determines the tracks the train travels on as well as the schedule. The authors refer to this as the pathing problem which is different from the routing problem considered in this work in terms of the scale considered. Although both problems relate to determining which track segment the train travels on, the pathing problem is concerned with the micro level and deals with, for example, which side of the track segment the train travels in a double-track segment or which train uses the siding at a meet or pass point. The routing problem is at a more macro level and determines which rail line the train travels when there are multiple lines serving the same origin/destination pair. This permits the consideration of larger scale train networks in the routing problem covering a larger area. We note, however, that in the literature the

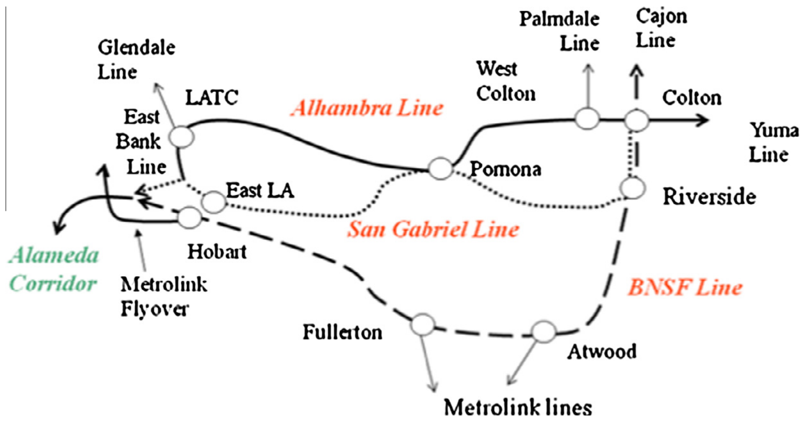


Fig. 1. Railway network between Downtown Los Angeles and Inland Empire trade corridor.

terms pathing and routing are used interchangeably. D'Ariano et al. (2008) propose iterative passenger train routing and sequencing procedures to support dispatchers in managing disturbances. They use a branch and bound algorithm for sequencing train movements, given train routes and then improve this solution by locally rerouting some trains. Unlike their work, we consider freight trains where routes and schedules are unknown. Additionally, we perform macro-level routing and scheduling.

There has been substantial prior work on train scheduling. Cordeau et al. (1998) and Caprara et al. (2006) present excellent surveys on this topic. Most of the research in railway scheduling has been done specific to passenger trains which, unlike freight trains, follow fixed timetables. Examples of train scheduling or timetabling include Cai and Goh (1994), Huntley et al. (1995), Higgins et al. (1996), Carey and Lockwood (1995), Hallowell and Harker (1998), Caprara et al. (2002), Dessouky et al. (2006), Zhou and Zhong (2007), D'Ariano et al. (2007, 2008), Lusby et al. (2011a, 2011b) and Cacchiani and Toth (2012).

There also has been a significant amount of work on the rescheduling problem and is also referred to as dispatching or the train conflict detection and resolution (CDR) problem in the literature (Corman et al., 2010) and normally involves passenger trains. The difference between the CDR problem and the freight train scheduling problem is that for the CDR problem, a schedule is given and the objective is to minimize the deviations from the schedule once disturbances occur in the network. Other papers on the train rescheduling and dispatching problem include the work of Kraay et al. (1991), Kraay and Harker (1995), Şahin (1999), Dorfman and Medanic (2004), Tornquist and Persson (2007), D'Ariano et al. (2007), Caimi et al. (2012), Boccia et al. (2013), Kecman et al. (2013), Corman et al. (2014), Meng and Zhou (2014), Pellegrini et al. (2014) and Cacchiani et al. (2014).

Although some of the papers may not assume the path of the train is given, most of the papers mentioned above present models that assume train routes to be known at a macro level, thus concentrating on train scheduling and dispatching alone. This paper represents an original effort in developing a quantitative model to tactically plan the movement of trains at the macro level through a complex network, with decisions based on an accurate representation of the delays these trains cause on the railroad. The routing problem is important since in many urban areas, like Southern California, there are several different rail routes. The capability to balance the freight rail traffic along the three routes has the potential to significantly reduce train travel times in the area.

We first formulate the joint routing and scheduling problem as an integer programming model (IP). The IP model assigns trains to routes that minimize travel times, in order to balance the traffic in a railway network. This model accounts for track capacities and speed-limits, train speeds, and also has deadlock avoidance constraints. Since the routing problem becomes interesting for medium to large-scale rail networks the IP formulation becomes too large to be solved directly, making it impractical to plan the movement of trains over real-world rail networks. To address this difficulty we consider an approximate problem obtained by aggregating sections of a network. The degree of aggregation to use requires the careful balancing between accuracy of the results and the computational difficulty of the resulting problem. The higher the degree of aggregation, the smaller is the size of the network, but lower is the accuracy of the model. This trade-off is explored in the sensitivity analysis section of this article. We also propose two solution procedures to solve the IP model: one based on a linear relaxation formulation of the IP, and the second based on routing constraints. These solution procedures are used to determine the routes and release times of trains from their origin stations. These are shown to play a major role in determining the travel times experienced in the network. We also develop a genetic algorithm procedure to improve the quality of the routes and release times generated by the first two solution procedures.

The remainder of this paper is organized as follows. In Section 2, we present an integer programming (IP) integrated routing and scheduling model, give examples on problem sizes for medium-scale networks, and present approximation methods to reduce the IP model size. In Section 3, we discuss approximation-based solution procedures to solve the integrated routing and scheduling for large, complex rail networks. Additionally, a genetic algorithm procedure is also developed to improve the quality of the routes and schedules generated by the solution methods. In Section 4, we present experimental results using multiple real-world and randomly generated test rail networks. The computational times of the recommended solution approach and its sensitivity to the degree of aggregation and traffic volumes are also presented. Finally, in Section 5, a few concluding remarks and opportunities for subsequent research are provided.

2. Model formulation

Given a railway network consisting of main tracks, sidings, junctions and platforms, it can be converted to a general network $G = (N,A)$, where N is the node set and A is the arc set. This representation has been used in Lu et al. (2004), Dessouky et al. (2006) and Mu and Dessouky (2011) among others. Each node $j \in N$ represents a portion of track segments of the rail network and contains a set of segment resources. We assume the length of node j is large enough that a train traveling on it can be completely contained in the segment. That is, the time spent by a train at a node j will be at least equal to the time needed for the head of the train to traverse the segment(s) included in that node. Additionally, we assume that trains are significantly smaller than the track segment length within the node. That is, we ignore that a train could be occupying multiple nodes at once. This means that the total travel time for a train can be represented as the sum of two quantities: (i) the travel time of the head of the train when traveling at the track speed limit and there are no additional trains present, and (ii) the wait time when additional trains are present and multiple meet/pass interactions occur between them. The capacity of these nodes depends on the number and type of track segments included in the node. The stations existing in the network are also represented by nodes. The capacity of these station-nodes is greater than 1, in order to accommodate crossing and overtaking. The nodes are connected to each other by directed arcs $a \in A$ which do not have any length or speed limit associated with them. There are two directed arcs, one for each direction, between any two geographically adjacent nodes i and j. To replicate the physical network, at most one of these two arcs can be occupied at a time. Similarly, each node j may be occupied by more than one train as long as the capacity of the segments constituting the node is not violated. It is important to note that the length of the track segment represented by a single node could impact the solution quality. For instance, a 100 mile long track could be represented by either a single node or split into, say, ten nodes of 10 miles each. However, there will be a trade-off between model accuracy and computational solution time when selecting the tracks within a node. Although the computational complexity is reduced significantly by combining multiple track segments into one aggregate node, there needs to be a method to accurately compute travel times for such nodes. We investigate the effect of aggregating very large sections of the network into fewer nodes on solution quality in Section 4.2.

For each train $h \in H$, the origin and destination stations are known, but the route can be either known or unknown. If the route is unknown for a train, the IP model would route and schedule the train. In this case, a train with unknown route is routed along the track segments (nodes) with the least travel time. On the other hand, if the route is known, then the IP model would just be a train scheduling model wherein the optimal release times from the origin stations are determined.

We define binary variables $Y_{i,j,h,t}$ to denote a train h traveling from node i to node j by time t We reduce the number of integer variables by defining Y only for possible directions of travel for each train. We assume that travel time is equal to the speed limit of the segment, assuming the train is traveling at constant speed. We also make an assumption that the trains have very high acceleration and deceleration rates to enable them to instantly change their speeds to follow the speed limits of the track segments they are traveling on. This assumption is important since acceleration and deceleration are quadratic functions, and we are interested at this point to keep the formulation computationally tractable. This allows us to compute travel time by simply dividing the length of the section by the track segment's speed limit. We take the solution from the optimization models that make these assumptions and feed them into a simulation model, explained in Section 4 of the paper, which does not make these assumptions. This enables us to evaluate the impact of the solutions made with these assumptions and see how they perform in a realistic setting. We consider a set of H trains traveling through a railway network. We assume that the direction of travel on the network separates the H trains in two subsets. For example, we can have a set of trains traveling North–South (or East–West) and another set traveling South–North (or West–East). All the routes available for a train will have the same direction of travel. To represent this, every node in the general network, described in the beginning of this section has two ports: port 0 and port 1. Port 0 indicates the starting point of travel for a train moving in the node from one direction. Port 1 indicates the starting point of travel in the opposite direction of port 0. If a train h enters a node from port 0 (respectively, port 1), then it must leave from port 1 (respectively, port 0). The set of directed arcs A represents these connections between nodes. Each node is connected to an adjacent node by two arcs to handle the two possible directions of travel. As a result, A comprises two subsets A_1 and A_2 that are defined below. We define the following parameters.

H_1	set of trains heading from '0' to '1'
H_2	set of trains heading from '1' to '0'
N	set of nodes representing track segments
A	set of directed arcs
A_1	set of arcs directed from '1' of the previous node to '0' of the next node
A_2	set of arcs directed from '0' of the previous node to '1' of the next node
v_i	velocity limit of node i
l_i	length of node i
c_i	capacity of node i
r_h	earliest departure (or release) time of train h
o_h	origin node of train h
d_h	destination node of train h
T	time horizon
ω_1	set of (A_1, H_1)
ω_2	set of (A_2, H_2)
Ω	$\omega_1 \cup \omega_2$
M	a very large number

The *decision variables* in our mathematical programming model are $Y_{i,j,h,t}$. These are variables defined such that $(i,j,h) \in \Omega$.

$$Y_{i,j,h,t} = \begin{cases} 1 & \text{if train } h \in H \text{ traverses } (i,j) \in A \text{ at any time between 0 and } t \\ 0 & \text{otherwise} \end{cases}$$

The integrated train routing and scheduling model is presented below. The objective function of this integer programming model is to minimize the sum of all arc traversal times for each train, thereby minimizing the total travel time for all trains. In the formulation below, weights can be associated with the travel time of each train to represent any priority that may exist. Here, we give an objective function assuming all trains are of equal importance.

$$\begin{aligned}(IP1)\quad & \text{Min} \quad \sum_{(i,j,h)\in\Omega}\sum_{t=1}^{T} t[Y_{i,j,h,t} - Y_{i,j,h,t-1}] \\ & \text{s.t.} \quad \text{constraints } (1)–(10) \\ & \qquad Y_{i,j,h,t} \in \{0,1\} \quad (i,j,h)\in\Omega,\ t\in\{1,\ldots,T\}\end{aligned}$$

where constraints ((1)–(10)) are explained below. Constraint (1) ensures that a train does not leave its origin station prior to its earliest departure time.

$$\sum_{(o_h,j)|(o_h,j,h)\in\Omega} Y_{o_h,j,h,r_h-1} = 0 \quad \forall h\in H \tag{1}$$

The following constraint (2) ensures that a train must eventually leave its origin station after the earliest departure time.

$$\sum_{(o_h,j)|(o_h,j,h)\in\Omega} [Y_{o_h,j,h,T} - Y_{o_h,j,h,r_h-1}] = 1 \quad \forall h\in H \tag{2}$$

The following constraint (3) enforces the condition that a train has to arrive at its destination station.

$$\sum_{(i,d_h)|(i,d_h,h)\in\Omega} Y_{i,d_h,h,T} = 1 \quad \forall h\in H \tag{3}$$

Constraint (4) enforces the condition that a train must take at least (length of segment/velocity limit over that segment) units of time to traverse the node representing that segment. Constraint (5) takes care of flow conservation, thereby ensuring that every train entering a node leaves the node after a certain amount of time.

$$\sum_{i|(i,j,h)\in\Omega} Y_{i,j,h,t} - \sum_{k|(j,k,h)\in\Omega} Y_{j,k,h,t+\left\lceil \frac{l_j}{v_j}\right\rceil} \geqslant 0 \quad \forall j\in N\setminus\{o_h,d_h\},\ h\in H,\ t\in\left\{0,1,\ldots,T-\left\lceil\frac{l_j}{v_j}\right\rceil\right\} \tag{4}$$

$$\sum_{i|(i,j,h)\in\Omega} Y_{i,j,h,T-\left\lceil \frac{l_j}{v_j}\right\rceil} - \sum_{k|(j,k,h)\in\Omega} Y_{j,k,h,T} = 0 \quad \forall j\in N\setminus\{o_h,d_h\},\ h\in H \tag{5}$$

Constraints (6) and (7) enforce node capacity constraints. In each time period t the number of trains traveling in node j in either direction must be less than c_j.

$$\sum_{(i,h)|(i,j,h)\in\omega_1} Y_{i,j,h,t} - \sum_{(k,h)|(j,k,h)\in\omega_1} Y_{j,k,h,t} \leqslant c_j \quad \forall j\in N,\ t\in\{0,1,\ldots,T\} \tag{6}$$

$$\sum_{(i,h)|(i,j,h)\in\omega_2} Y_{i,j,h,t} - \sum_{(k,h)|(j,k,h)\in\omega_2} Y_{j,k,h,t} \leqslant c_j \quad \forall j\in N,\ t\in\{0,1,\ldots,T\} \tag{7}$$

Constraints (8) and (9) ensure that trains traveling in opposite directions cannot enter the same node at the same time. If a node is already occupied by a train h, then a train h' traveling in the opposite direction would either need to wait for this node to be freed or may choose an alternative node to reach its destination. Trains traveling in the same direction can occupy the same node at the same time, provided that the capacity constraint for that node is not violated.

$$\sum_{(i,h)|(i,j,h)\in\omega_2} Y_{i,j,h,t} - \sum_{(k,h)|(j,k,h)\in\omega_2} Y_{j,k,h,t} \leqslant M\left[1-\sum_{k|(k,j,h')\in\omega_1}\left(Y_{k,j,h',t} - Y_{k,j,h',t-1}\right)\right] \quad \forall j\in N\setminus\{o_{h'},d_{h'}\},\ h'\in H,\ t\in\{1,\ldots,T\} \tag{8}$$

$$\sum_{(i,h)|(i,j,h)\in\omega_1} Y_{i,j,h,t} - \sum_{(k,h)|(j,k,h)\in\omega_1} Y_{j,k,h,t} \leqslant M\left[1-\sum_{k|(k,j,h')\in\omega_2}\left(Y_{k,j,h',t} - Y_{k,j,h',t-1}\right)\right] \quad \forall j\in N\setminus\{o_{h'},d_{h'}\},\ h'\in H,\ t\in\{1,\ldots,T\} \tag{9}$$

Constraint (10) ensures that an arc is not simultaneously occupied by trains traveling in the opposite direction, since this could lead to a deadlock.

$$\sum_{h\in H|(i,j,h)\in\Omega} (Y_{i,j,h,t} - Y_{i,j,h,t-1}) + \sum_{h\in H|(j,i,h)\in\Omega} (Y_{j,i,h,t} - Y_{j,i,h,t-1}) \leqslant 1 \quad \forall t\in\{1,\ldots,T\},\quad \forall (i,j)\in\{A_1|(j,i)\in A_2\} \tag{10}$$

We note that the model above can consider trains that have different lengths and speeds. For this, in Eqs. (4) and (5), we would need to make the speed variable indexed in the speed limit of the train and that of the node.

To address real-world railway routing and scheduling, we need to be able to run our (*IP*1) model for medium-scale networks, which are 30–60 miles long, with a relatively short computational time. That is, we need to do train routing and scheduling at a macro-level efficiently. Solving the integer programming model presented above for medium to large-scale real-world rail networks is beyond the ability of current solvers and computing power. In Fig. 1, dividing each of the three routes into nodes by considering the length of the longest train traveling on these lines, we get approximately 40 nodes per route. If we solve the formulation (*IP*1) for this network with a daily traffic of 50 trains, then we end up with approximately 200,000 binary variables and 350,000 constraints. Note that the number of binary variables and constraints increases drastically with the addition of new nodes. Thus, this current formulation (*IP*1) inhibits our ability to solve the routing and scheduling problem for medium to large-scale rail networks. In the interest of being able to solve IP models we resort to an approximation of the problem done by aggregating parts of the graph, this procedure, referred to here as Aggregation, helps reduce the number of nodes and arcs in the general network.

2.1. Aggregation

Aggregation refers to combining a sizeable portion of the network under consideration into a single node in the graph *G*. An example of aggregation is presented using Fig. 2, which is a section of the railway network near Los Angeles.

For example, consider the track portion between A and B in Fig. 2. This segment contains 7 different track segments that can contain the longest train (all the horizontal lines) and three possible connections between these double tracks (the diagonal lines). This part of the network is represented in (IP1) using 7 nodes and eight arcs, while an aggregate representation would use a single node for the section AB but require a more sophisticated representation of train delay and capacity on the node. The physical network shown in Fig. 2 above between A and D, with start and end nodes, can be represented in (IP1) with a network with 15 nodes and 21 arcs without aggregation, however it would only require 5 nodes and 4 arcs by aggregating sections of this network. We refer to a network as aggregated when some nodes consider more than a single train track segment. Clearly, aggregation helps to reduce the problem size by decreasing the number of nodes and arcs in *G* for a given physical network. We follow some fundamental guidelines for aggregation – each node can have only similar type of track segments. We cannot combine single, double and triple-track segments into a single node. In Fig. 2, AB, BC and CD are double-track, single-track and double-track sub-networks respectively. So, based on the aforementioned guidelines, we aggregate these into 3 nodes as shown in the figure.

For networks without aggregation, we used a node capacity of 1 in the (*IP*1) formulation. Also, the time taken to travel through a node was assumed to be linear, and equal to the length of the node divided by the speed limit of that node. During aggregation since many smaller nodes are combined into a bigger node, we need to reevaluate the capacity of an aggregate node. An aggregated node could accommodate trains traveling in opposite directions. In an aggregate node, the assumption of instantaneous acceleration and deceleration is less valid because of a higher probability of trains meeting and overtaking each other. Hence, the time taken to travel through a node would no longer be linear. Moreover, trains might need to wait for each other before they exit one aggregate node and enter another. In view of these reasons, a more detailed and robust capacity and travel time estimation procedure needs to be developed.

In the routing and scheduling IP model described above, the only property of the individual nodes that was considered to estimate the capacity of an aggregated node was the length of the track segments. Track configurations, interactions between

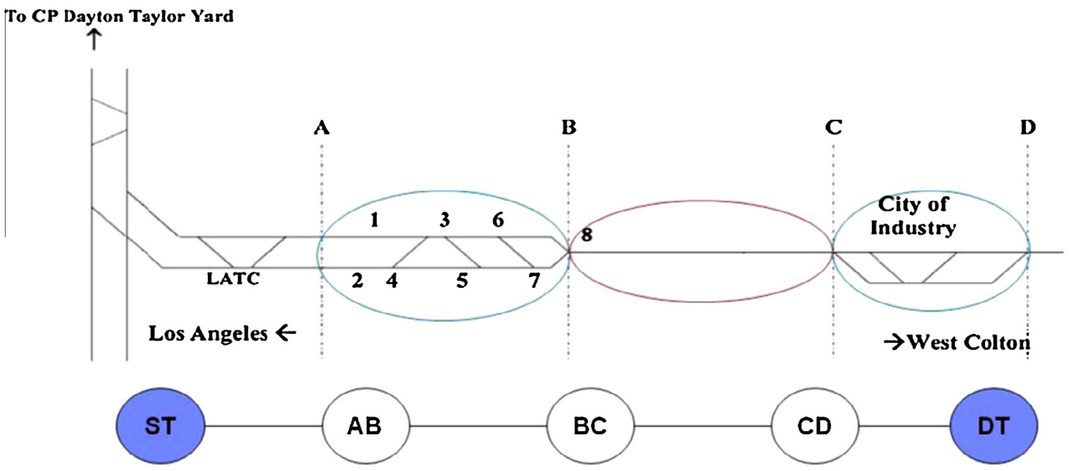


Fig. 2. An example of aggregation.

trains and knock-on delay effects were not explicitly considered. Hence, the actual travel time of the trains, from their origins to their respective destinations, might be quite different from the values obtained from the IP model. Some key aspects of real-life railway operations that are taken into account in the aggregate strategy proposed include the following:

1. Nodes N in G that make up an aggregate node j can have different capacities c_i and speed-limits v_i. Our strategy presented above computes c_j for each $j \in N_a$, the set of aggregate nodes, in a simplified fashion, without considering the variations in the speed limits of the track segments and the trains.
2. At a given point in time, the nodes constituting an aggregate node j can have multiple trains traveling on the track segments they represent. In addition, these trains could be traveling in opposite directions.
3. The travel time across an aggregate node j is dependent upon the physical characteristics of the track segments constituting j. The type of trackage (single-track, double-track, etc.), the number of meet–pass points, spacing between the meet–pass points, track speed-limit and the number of sidings need to be considered.
4. The travel time across an aggregate node j is dependent upon the current traffic in j. That is, the number of trains of various types, their direction of travel and speed-limits.

To represent these aspects of train operations we develop a travel time estimation technique that accounts for the complexities and non-linearities that exist in real-world railway operations. Instead of the simple aggregation step presented previously, we use a regression-based technique that accurately estimates the travel time for each aggregate node j as a function of the track configuration of the nodes in N that form the aggregate node j, as well as the prevailing traffic in j. The capacity of each j can be set such that the time taken by an incoming train to traverse j is within travel time thresholds. The travel estimation procedure is explained in detail in Murali et al. (2010). The authors show that the natural logarithms of the travel time can be expressed as a linear function of (1) the number of trains of each type in the aggregate node (i.e., it is a function of train type since long slow freight trains will have a larger impact on other trains traveling times than fast passenger trains), (2) the total number of trains traveling in the opposing direction currently in the aggregate node, and (3) the total number of trains that will enter the node in the opposite direction relative to the train whose travel time is being estimated. We note that, in order to maintain a mixed integer programming formulation for the integrated routing and scheduling model, we constrain ourselves to linear regression-based equations that capture the dependencies between these same parameters and travel time.

For each aggregate node j, we denote $\Delta1_{j,t}$ and $\Delta2_{j,t}$ as the travel time in the two possible directions for a train starting its travel on aggregate node j at time t. Note that $\Delta1_{j,t}$ and $\Delta2_{j,t}$ are indexed in t because they are a function of the prevailing traffic and operating parameters in aggregate node j, both of which vary with time. The values $\Delta1_{j,t}$ and $\Delta2_{j,t}$ are determined by a relation to the number and types of trains currently in the node and in the future. These travel times are a function of three components:

$X_{p,j,t}$	the number of trains of type p in aggregate node j at time t
$F_{d,j,t}$	the number of trains that can enter the sub-network in the opposite direction to d while the train under consideration is in j
$G_{d,j,t}$	the number of trains traveling in the opposite direction to d that have entered j by t, but have not yet departed from j

We approximate $F_{d,j,t}$ by expressing it as the number of trains traveling in the opposite direction in nodes adjacent to j, that is in all nodes k such that $(k,j) \in A$, at time t. Let P denote the set of different types of trains (lengths, speeds, freight type, etc.). Then the following linear relationships relate the travel time at aggregate node j at time t to these train counts. The coefficients of these linear equations are tuned using linear regression on simulated travel time data for networks with the configuration of the aggregated node.

$$\Delta1_{j,t} = \alpha1_j + \sum_{p \in P} c_{p,j} X_{p,j,t} + f_{1,j} F_{1,j,t} + g_{1,j} G_{1,j,t} \quad \forall j \in N_a,\ t \in \{1, \ldots, T\} \tag{11}$$

$$\Delta2_{j,t} = \alpha2_j + \sum_{p \in P} c_{p,j} X_{p,j,t} + f_{2,j} F_{2,j,t} + g_{2,j} G_{2,j,t} \quad \forall j \in N_a,\ t \in \{1, \ldots, T\} \tag{12}$$

Here, $\alpha1_j$ and $\alpha2_j$ are the constant terms in the regression equation and represent the free flow running time of a train. We use coefficients $c_{p,j}$, $f_{1,j}$, $f_{2,j}$, $g_{1,j}$ and $g_{2,j}$ obtained from the regression-based travel time function. Let $\Omega_p \subset \Omega$ only consider trains of type p. Recall that $\Omega = \omega_1 \cup \omega_2$ where ω_1 consist of trains that are traveling in direction 1 and ω_2 consist of trains that are traveling in direction 2. Eqs. (13)–(17) help define the quantities that are relevant for the travel time linear regression as a function of the problem variables $Y_{i,j,h,t}$. For instance Eq. (13) states that the number of trains of type p that are in node j at time t is given by the sum of all the trains of type p that have entered j by time t minus all that have left j by that time.

$$X_{p,j,t} = \sum_{i,h|(i,j,h)\in\Omega_p} Y_{i,j,h,t} - \sum_{k,h|(j,k,h)\in\Omega_p} Y_{j,k,h,t} \quad \forall j \in N_a,\ p \in P \tag{13}$$

$$F_{1,j,t} = \sum_{k|(k,j)\in A_2} \left(\sum_{l,h|(l,k,h)\in\omega_2} Y_{l,k,h,t} - \sum_{m,h|(k,m,h)\in\omega_2} Y_{k,m,h,t} \right) \quad \forall j \in N_a,\ t \in \{1,\dots,T\} \tag{14}$$

$$F_{2,j,t} = \sum_{k|(k,j)\in A_1} \left(\sum_{l,h|(l,k,h)\in\omega_1} Y_{l,k,h,t} - \sum_{m,h|(k,m,h)\in\omega_1} Y_{k,m,h,t} \right) \quad \forall j \in N_a,\ t \in \{1,\dots,T\} \tag{15}$$

$$G_{1,j,t} = \sum_{k,h|(k,j,h)\in\omega_2} Y_{k,j,h,t} - \sum_{i,h|(j,i,h)\in\omega_2} Y_{j,i,h,t-1} \quad \forall j \in N_a,\ t \in \{1,\dots,T\} \tag{16}$$

$$G_{2,j,t} = \sum_{k,h|(k,j,h)\in\omega_1} Y_{k,j,h,t} - \sum_{i,h|(j,i,h)\in\omega_1} Y_{j,i,h,t-1} \quad \forall j \in N_a,\ t \in \{1,\dots,T\} \tag{17}$$

With $\Delta 1_{j,t}$ and $\Delta 2_{j,t}$ we can now express the travel time constraints for the aggregate nodes. We first define a new variable $Z_{i,j,h}$ which equals the time train h enters j from i if $Y_{i,j,h,t}$ is greater than zero, and it is equal to zero otherwise. Let

$$Z_{i,j,h} = \sum_{t=1}^{T} t * [Y_{i,j,h,t} - Y_{i,j,h,t-1}] \quad \forall j \in N_a,\ (i,j,h) \in \Omega \tag{18}$$

Constraints (19) and (20) enforce the minimum time a train entering an aggregate node j takes to travel through it, as per the regression-based travel time equations.

$$\sum_{k|(j,k)\in A_1} Z_{j,k,h} - \sum_{i|(i,j)\in A_1} Z_{i,j,h} + \left[1 - \sum_{i|(i,j)\in A_1} (Y_{i,j,h,t} - Y_{i,j,h,t-1})\right] * M \geqslant \Delta 1_{j,t} \quad \forall h \in H, \quad \forall j \in N_a,\ t \in \{1,\dots,T\} \tag{19}$$

$$\sum_{k|(j,k)\in A_2} Z_{j,k,h} - \sum_{i|(i,j)\in A_2} Z_{i,j,h} + \left[1 - \sum_{i|(i,j)\in A_2} (Y_{i,j,h,t} - Y_{i,j,h,t-1})\right] * M \geqslant \Delta 2_{j,t} \quad \forall h \in H, \quad \forall j \in N_a,\ t \in \{1,\dots,T\} \tag{20}$$

In summary, the integrated train routing and scheduling model with aggregated nodes is the previous (IP1) problem, to which we incorporate constraints (11)–(20) for aggregated nodes and variables X, Z, F, G, $\Delta 1_{j,t}$ and $\Delta 2_{j,t}$:

$$\begin{aligned}
(IP2) \quad \text{Min} \quad & \sum_{(i,j,h)\in\Omega} \sum_{t=1}^{T} t[Y_{i,j,h,t} - Y_{i,j,h,t-1}] \\
\text{s.t.} \quad & \text{constraints } (1)-(20) \\
& Y_{i,j,h,t} \in \{0,1\} \quad (i,j,h) \in \Omega,\ t \in \{1,\dots,T\} \\
& X_{p,j,t} \geqslant 0 \quad J \in N_a,\ p \in P,\ t \in \{1,\dots,T\} \\
& Z_{i,j,h} \geqslant 0 \quad (i,j,h) \in \Omega \\
& F_{1,j,t}, F_{2,j,t}, G_{1,j,t}, G_{2,j,t} \geqslant 0 \quad J \in N_a,\ t \in \{1,\dots,T\}
\end{aligned}$$

3. Approximation-based solution procedures

Using the integer programming model ($IP2$) presented in Section 2 together with the aggregation and travel time estimation techniques, we can now tackle the problem of routing and scheduling trains on large complex rail networks with combinations of track segments of varying lengths, speed-limits, traffic conditions, and number of crossings, junctions and sidings. The usual approaches used by researchers in the past to solve large-scale integer programming problems include exact solution methods such as cutting planes and branch-and-cut methods, and approximation procedures. Due to the size of the problem at hand, we use approximation-based solution procedures to solve ($IP2$) for real-world sized rail networks.

At the planning level, some of the important decisions for the freight trains are the identification of the route for each train and the departure times at the origin stations. Hence, our solution procedures focus on identifying suitable integer solutions based on the LP relaxation formulation of ($IP2$) for these two decisions. The decisions related to the detailed sequence of trains at the subsequent stations in the network can be made later at the operational level. Our proposed procedures determine only the route that each train takes and the order of departure at the origin of each train with an objective to reduce the number of meet–pass interferences down the line.

3.1. A solution approach using LP-relaxation

One solution approach is to relax the integrality constraints on the $Y_{i,j,h,t}$ binary variables, obtain a solution to this relaxed problem, and then construct a feasible integer solution from this fractional solution. Since ($IP2$) is a minimization problem, its linear relaxation affords a lower bound to the original problem. This lower bound solution allows each train h to be *split* across the various possible arcs (o_h,j) from its origin o_h and across different time intervals t. Each arc (o_h,j) is part of a possible route between the origin and destination of train h. A simple rounding algorithm that allocates a train h to a route corresponding to the largest $Y_{o_h,j,h,t}$ value is applied to the solution obtained from the LP-relaxation problem. Another piece of information derived from the LP-relaxation output is the order of departure from the origin nodes. We solve the following maximal matching problem (Papadimitriou and Steiglitz, 1998) to decide the departure times for each train from their respective origins.

$$
\begin{aligned}
(P_1)\quad \text{Max} \quad & \sum_{h\in H, t\in\{1,\ldots,T\}} \beta_{h,t}\varphi_{h,t} \\
\text{s.t.} \quad & \sum_{h\in H}\varphi_{h,t} = 1 \quad \forall t\in\{1,\ldots,T\} \\
& \sum_{t\in\{1,\ldots,T\}} \varphi_{h,t} = 1 \quad \forall h\in H \\
& \varphi_{h,t}\in\{0,1\} \quad \forall h\in H,\ t\in\{1,\ldots,T\}
\end{aligned}
$$

In the above formulation, $\varphi_{h,t} = 1$ if train h departs its origin at time t, else it is 0. $\beta_{h,t} = Y_{o_h,j,h,t} - Y_{o_h,j,h,t-1}$ such that (o_h,j, h) $\in \Omega$. The first constraint enforces the condition that only one train can depart at any time instant t from the origin station, and the second constraint ensures that train h departs only once over the planning time horizon. The solution obtained after applying the rounding algorithm and solving the matching problem is still not a feasible solution for ($IP2$), but it completely characterizes the initial decisions: which routes are taken by each train and what are their scheduled departure times. We call this solution *routing and release from LP-relaxation*, abbreviated as "R&R_LP". In summary, Procedure "R&R_LP" only provides feasible routing and release schedules but not a complete schedule for all the nodes in the network. Given these routing and release schedules, a simple greedy conflict avoidance heuristic, as in (Lu et al., 2004), can be implemented to obtain a complete schedule for the trains to travel through the entire rail network.

3.2. A solution approach using routing constraints

Another approach to obtain a solution is to relax all variables $Y_{i,j,h,t}$ except those out of the origin o_h of every train h. This however repeats the original movement decision over all time periods. To reduce the number of integer variables considered we introduce new binary variables $zz_{o_h,j,h}$ defined for all origin nodes o_h for $h \in H$ The variable $zz_{o_h,j,h}$ is equal to 1 if train h travels to node j from its origin node o_h, and is equal to 0 otherwise. The routing constraints are:

$$Y_{o_h,j,h,t} \leqslant zz_{o_h,j,h} \quad \forall j|(o_h,j,h)\in\Omega,\ h\in H \tag{21}$$

$$\sum_{j|(o_h,j,h)\in\Omega} zz_{o_h,j,h} \leqslant 1 \quad \forall h\in H \tag{22}$$

Constraint (22) enforces the condition that for a given train h, only one node j such that (o_h,j,h) $\in \Omega$ will be chosen as a successive node when h departs its origin. Constraint (18) assigns an integral value to the Y variable corresponding to arc (o_h, j) along which train h moves. The Y variables corresponding to the remaining forward arcs from o_h are forced to be zero. Similar routing constraints can also be added for intermediate nodes from where trains could depart along multiple routes. Adding constraints (21) and (22) to the LP-relaxation of ($IP2$) results in a *mixed integer linear program* (MILP) with discrete and continuous decision variables, $zz_{i,j,h}$ and $Y_{i,j,h,t}$ respectively.

The solution to the order of departure of trains from their origins can be determined once again by solving the matching problem (P_1). The solution obtained by solving ($MILP$) and the matching problem is again not feasible for ($IP2$), but gives initial routing and release scheduling decisions that can be implemented. We call this solution *routing and release from MIP*, abbreviated as "R&R_MIP".

3.3. A genetic algorithm (GA) solution procedure

The two previously presented approximation-based solution procedures entail using a rounding algorithm or a MILP with routing constraints and a matching problem to obtain routing and release schedules. One possible way to improve upon the quality of the initial routes and release schedules is to use a search algorithm that explores the solution space using a suitable evaluation criterion. Genetic algorithm is one such procedure that has been applied in the past to solve large-scale scheduling and timetabling problems. Prior work that have used genetic algorithms for railway operations include Nachtigall and Voget (1996, 1997), Salim and Cai (1996, 1997) and Suteewong (2006) among others. For a good survey of the use of GA to solve mixed integer programs, especially scheduling problems see the book by Gen and Cheng (2000).

Each chromosome in our GA is represented by the binary variables $\Gamma_{i,j,h,t}$. This variable, defined as $\Gamma_{i,j,h,t} = Y_{i,j,h,t} - Y_{i,j,h,t-1}$ for any $(i,j,h) \in \Omega$, $t \in \{1,\ldots,T\}$, is equal to 1 if train h travels from node i to node j at time t, and 0 otherwise. We define matrices for every combination of an origin station for train $h \in H$, and initial arc traversed from that origin node. The rows of the matrix represent the set of trains, and the columns represent the time of departure. If train h departs from its origin o_h along arc (o_h,j) at time t, then the corresponding block in the matrix that represents arc (o_h,j) will be marked 1, otherwise it will be marked 0. The chromosomes represent routes to the destination through the arc chosen by a train when it departs from its origin. In the ensuing discussion, we use the terms "routes" and "arcs" interchangeably. A detailed explanation of the selection, crossover, mutation and repairing operations of the GA is provided in Murali (2010). Given below is the outline of the heuristic.

1. Given an initial departure time information (from either R&R_LP or R&R_MIP), translate this solution into its genetic representation.
2. Use the initialization step to generate the initial population of chromosomes representing the $\Gamma_{i,j,h,t}$ binary variables for the origin stations.
3. Compute the travel time associated with each chromosome using the construction heuristic developed by Lu et al. (2004). It should be noted here that chromosomes reflect only the routes and departure times from the origin station. These are fed into the construction heuristic which then schedules trains at the intermediate stations. The travel time values obtained from the construction heuristic reflect the quality of the routes and schedules at the origin station. Good quality solutions minimize the meet–pass interferences between trains at intermediate stations and junctions, thereby minimizing travel time.
4. Use roulette wheel selection strategy to select parent chromosomes for the reproduction process.
5. Perform crossover and mutation operations and use their respective repairing algorithms to rectify infeasibilities in the children chromosomes generated by the reproduction process.
6. Step (5) generates a new population for the $\Gamma_{i,j,h,t}$ binary variables. Compute the travel time values for the new population similar to step (3).
7. Check the termination criteria. If this is met, terminate the algorithm and output the values of the $\Gamma_{i,j,h,t}$ variables corresponding to the current best chromosome. If not, return to step (5). The termination criterion in our case is when there is no significant improvement in the travel time values between two consecutive iterations.

3.4. Overall solution approach

We have presented four different heuristic approaches (R&R_LP, R&R_MIP, R&R_LP_GA, and R&R_MIP_GA) to obtain routing and release schedule solutions. To obtain a complete schedule for all the nodes in the network, the routing and release schedule decisions are fed into a simulation model (see Lu et al., 2004) which includes many of the actual characteristics of a rail network (e.g., capabilities to model train lengths and the acceleration and deceleration process). The simulation model uses a construction heuristic to schedule the trains for all the intermediate nodes. A summary of the overall solution procedure is outlined below. Details of steps 1 and 3 can be found in Murali et al. (2010) and in Lu et al. (2004), respectively. The contribution of this paper is in step 2.

1. Run simulations over every aggregate node individually by varying traffic volumes, arrival rates and the mix of the various types of trains. Run regression over the collected data to generate the travel time estimation equations. Formulate the travel time constraints (11) and (12), and add them to ($IP2$) along with Eqs. (13)–(15).
2. Follow the steps detailed for solution procedures *R&R_LP* and *R&R_MIP*. By treating the routes and release times information generated by the *R&R_LP* and *R&R_MIP* solution schemes as parent chromosomes, a genetic algorithm procedure is used to improve the quality of these solutions. These are called *R&R_LP_GA* and *R&R_MIP_GA* respectively.
3. Feed the routing and release schedule decisions obtained from one of the procedures in step 2 to a simulation model. The construction heuristic within the simulation develops a deadlock-free schedule for the trains when they travel through the intermediate stations. Record the travel time values from the output of the simulation model.

4. Experimental results

The first set of experiments (Section 4.1) are used to test the performance of the heuristics based on actual train networks in the Los Angeles area using aggregation levels where the problem is computationally solvable while the second set of experiments (Section 4.2) then test the sensitivity to the degree of aggregation using the best heuristic found in Section 4.1. We use smaller network sizes in Section 4.2 since we tested cases where no aggregation is performed which is not computationally feasible for the larger networks used in Section 4.1. The structure of these networks allows varying the degree of aggregation.

4.1. Comparison of heuristics' performance

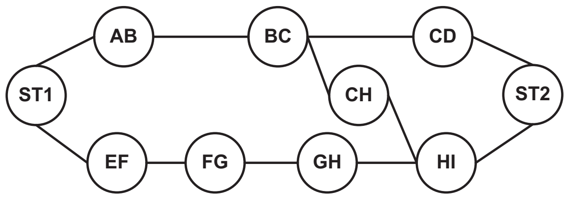
To test the performance of the various heuristics, we consider an instance of the LA network and four more test networks that are constructed using values of track segment length in miles (L), speed-limit in miles per hour (V), number of crossings or sidings (C) and spacing (S) similar to the typical characteristics of the rail network in the Southern California region. If it is the double-track segment (C) represents the number of crossings in the segment and if it is a single-track segment it represents the number of sidings. The spacing (S) is defined as the portion of the subnetwork over which crossings (or sidings) are uniformly distributed. If crossings are uniformly distributed over the entire track length (i.e., $S = 1$), then trains can more easily overtake and/or cross each other, than if all the crossings are concentrated at one end of the network segments. Therefore, travel time increases with a decrease in S due to possible interactions between trains on two consecutive crossings (or sidings).

The LA rail network used for our experiments is between Alameda Corridor and Pomona, as shown in Fig. 1. We consider two routes available for trains beginning at either end – UP-Alhambra line and UP-San Gabriel line. Each route is approximately 30 miles long. We note the test instance of Los Angeles is a real-world instance, as are the trains considered in this paper. For further information on this network, we refer the reviewer to Lu et al. (2004). The daily arrival rate of trains is close to the actual values but we modified them to ensure the networks were not overly congested or lightly congested in order to test the performance of the various heuristics. We run experiments with a sizeable number of trains such that there is a possibility of congestion being induced in network sections with crossings and sidings.

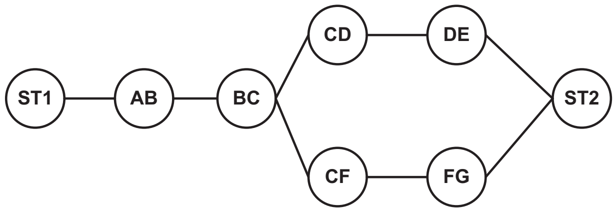
Next, we fix the skeleton networks, which are general networks with a given number of nodes and arcs. The two skeleton networks that we use for our experiments are shown in Fig. 3. In Skeleton Network 1, routes need to be determined for origin nodes ST1 and ST2 and intermediate aggregate nodes BC and HI since these are the decision points where a train can choose a particular route to travel. The initial release schedules at ST1 and ST2 also need to be determined. The link BC-CH-HI induces interaction between trains traveling in opposite directions. In Skeleton Network 2, routes need to be determined for origin node ST2 and intermediate node BC, and release schedules for origins ST1 and ST2. Since there is a single route between ST1 and BC, the quality of the release schedules from ST1 is vital to minimize travel times and interference between trains. The network characteristics (length, speed-limit, number of crossings/sidings and spacing between crossings/sidings) for aggregate nodes CD, DE, CF and FG are set so that it is suboptimal for all trains moving in the same direction to travel on the same route between BC and ST2. This would induce interaction between trains traveling in opposite directions. The time horizon for all experiments is one day.

Skeleton Network 1 has 10 nodes including 2 stations, and Skeleton Network 2 has 8 nodes including 2 stations. All nodes, except ST1 and ST2, are aggregate nodes. For our experiments, we generate four test networks. The configuration of the aggregate nodes in each of these networks is listed in Table 1. Networks 1, 2, and 3 are based on Skeleton Network 1 from Fig. 3. These networks differ in the length of the track segments (L), speed of the trains (V), number of crossings/sidings (C), and spacing (S). The settings for the Network 2 will result in a more congested rail system than Network 1 while Network 3 will result in the largest travel times. Network 4 is based on Skelton Network 2 from Fig. 3. We note a "D" under the number of tracks refers to double-track segments and an "S" refers to single-track segments.

Skeleton network 1 is representative of a portion of the LA network shown in Fig. 1. Skeleton network 2 considers less alternative routes between ST1 and ST2 than skeleton network 1. In Table 1, network 2 is longer compared to network 1,



Skeleton Network 1



Skeleton Network 2

Fig. 3. Experimental design: skeleton networks.

Table 1
Parameters for experimental test networks.

Node	# of tracks	*L*	*V*	*C*	*S*
Network 1					
AB	Double	12.5	35	3	1
BC	Single	10	55	2	1
CD	Single	10	55	2	1
CH	Double	5	35	1	1
EF	Double	5	35	1	1
FG	Single	10	55	3	1
GH	Double	12.5	35	1	1
HI	Double	5	35	1	1
Network 2					
AB	Double	12.5	35	3	1
BC	Single	10	55	2	1
CD	Single	10	55	2	1
CH	Double	5	35	1	1
EF	Double	12.5	15	3	0.5
FG	Single	15	35	4	0.7
GH	Double	20	35	1	0.5
HI	Double	12.5	35	5	0.75
Network 3					
AB	Single	10	35	3	0.85
BC	Double	20	55	3	1
CD	Single	15	15	4	1
CH	Single	15	35	2	0.7
EF	Double	5	35	1	1
FG	Single	10	35	3	0.85
GH	Single	15	15	4	1
HI	Single	10	35	3	0.85
Network 4					
AB	Double	5	35	3	1
BC	Double	12.5	55	3	1
CD	Double	12.5	55	3	1
DE	Double	5	35	3	1
CF	Single	15	35	3	1
FG	Single	20	35	3	0.85

Table 2
Problem sizes.

Network	Train count	Binary var.	Constraints
LA network	80	94,000	97,000
Network 1	80	101,000	86,000
Network 2	56	55,000	49,000
Network 3	80	66,000	63,000
Network 4	56	38,000	44,000

but has a lower speed limit which allows us to compute travel times under increased network congestion. Again, in comparison to network 1, network 3 introduces a structural change wherein FG and HI are now single track segments. This increases delays in the network. Compared to networks 1, 2 and 3, network 4 limits flexibility and routing to only one half of the network.

For our experiments, instances of (*IP*2), its LP-relaxation and its variants were run on a 3.06 GHz Intel Xeon CPU for 2.0 h. The integer and linear programs were formulated in AMPL and solved with IBM ILOG's CPLEX Optimizer version 12.0. The simulation model and its underlying construction heuristic were run using the AweSim! Simulation software (Pritsker and O'Reilly, 1999).

Table 2 shows the sizes of the formulation (*IP*2) for the different networks under different daily train counts. For these problem sizes, CPLEX is unable to find an optimal solution to (*IP*2). However, since *R&R_LP* and *R&R_MIP* generate relaxed solutions to (*IP*2), we were able to find solutions to these problems.

Table 3
Comparison of average travel times (min).

Network	Greedy	R&R_LP	R&R_LP_GA	R&R_MIP	R&R_MIP_GA	CPLEX rounding	CPLEX best
LA rail network	292.85	246.32	244.64	239.74	235.10	241.74	–
Network 1	90.06	83.11	82.95	80.69	78.48	81.90	–
Network 2	144.90	141.17	140.91	138.14	137.97	140.34	304.19
Network 3	256.82	215.19	213.08	212.36	209.67	214.61	498.71
Network 4	142.72	134.20	133.19	131.05	130.72	132.29	284.08

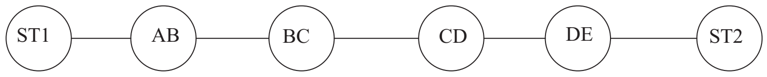


Fig. 4. Skeleton structure for test network 5.

We test the quality of the routes and release schedules obtained from the optimization model by generating a complete solution that includes train schedules from the intermediate nodes. The complete solution is generated for each solution procedure (*R&R_LP*, *R&R_LP_GA*, *R&R_MIP*, and *R&R_MIP_GA*) and the lower and upper bounds to (*IP*2) generated by CPLEX after 2 h of CPU time. At the end of the runtime CPLEX provides the best integer solution found, referred to as CPLEX Best, and the best lower-bound LP relaxation, which can be rounded to obtain another feasible solution to problem (*IP*2), referred to as CPLEX Rounding. The routes and initial release schedules obtained in each case are then input into the simulation model which includes many of the actual characteristics of a rail network (e.g., capabilities to model train lengths and the acceleration and deceleration process). For the intermediate nodes, all solutions use the construction heuristic for scheduling the trains. We record the average travel times for trains traveling on the network from the simulation model associated with the complete solution in Table 3. The performance of our solution approaches is also compared to a greedy approach for determining routes and initial release schedules. For the greedy approach, we route trains by balancing traffic volume on the various possible routes. That is, a train is assigned to a route with the least assigned traffic so far. Then, a greedy heuristic determines the initial release schedules once the routes are provided to it.

From Table 3 we can infer that choosing the right routes plays an important role in reducing travel times. Our solution procedures *R&R_LP*, *R&R_MIP* and the GA procedure all perform better than the greedy heuristic for all networks mainly because the initial routes and release schedules are determined using (*IP*2). Since the main difference between our solution procedures and the greedy heuristic is the initial routes and release times, the results shown in Table 3 attest to the importance of determining these parameters. The next observation is that the GA procedure reduces travel times for both the solution procedures. For *R&R_MIP_GA*, with the routes being determined optimally, nearly all the improvement in the travel times can be attributed to finding a better initial release schedule. The travel times obtained by using the routes and release times from CPLEX Rounding solution, were found to be worse-off than *R&R_MIP*, *R&R_LP_GA* and *R&R_MIP_GA*. This is because most of the $Y_{i,j,h,t}$ variables were fractional in the solution to the best lower bound, and the routes and schedules were determined by using a simple rounding algorithm. On the other hand, the upper bound to (*IP*2) was an integer solution, but was not a tight upper bound. The traffic was not balanced across the routes which resulted in opposing trains being routed along the same lines. In the simulation model, this translates to higher congestion arising from a large number of meet–pass interferences between the trains. This results in dramatically high travel times in the column representing the CPLEX Best solution. Furthermore, CPLEX could not find an upper bound to (*IP*2) after 2 CPU hours for the first two networks in the table. The final observation that we can make from the above table is that the best solution procedure is *R&R_MIP_GA*. The improvement in travel time obtained by this solution procedure over the solution found by the *Greedy* procedure on the LA Network corresponds to a (292.85–235.10)/292.85 = 20% improvement in travel time.

4.2. Sensitivity to aggregation

A key approximation procedure in our modeling approach is aggregation. As we increase the size of the network section aggregated into a single node (hereafter referred to as the *degree of aggregation*), we lose accuracy in terms of the optimization model, because the estimated travel time equations become less representative of the actual travel time. This, in turn, affects the quality of the routes and release times obtained from the optimization model using one of the aforementioned solution procedures.

For this purpose, we conduct sensitivity analysis tests. First, using skeleton structures we build test networks comprising of single-track and double-track sections. Next, for each network we vary the degree of aggregation by varying the length of the network section represented by each aggregate node in the general network. This translates to varying the number of nodes and arcs in $G(N,A)$. For each level of aggregation, we vary the train traffic volume over the network and obtain travel time values and solution times by using our recommended solution approach from the previous section, *R&R_MIP_GA*. For this sensitivity analysis we use smaller test rail networks, in terms of total length, than before since for the base case we set the aggregate node lengths to be the maximum train length (e.g., 1.5 miles). In this manner, we can test the sensitivity of the aggregation to the base case which requires no aggregation.

Table 4
Parameters for test network 5.

Node	# of tracks	*L*	*V*	*C*	*S*
AB	Double	5	35	1	1
BC	Double	12.5	35	3	1
CD	Single	15	15	3	1
DE	Single	5	15	2	1

Table 5
Travel time for test network 5 without aggregation.

	# of nodes	# of arcs	# of trains	CPU time (s)	Travel time (min)
Test network 5	31	27	4	210.58	56.31
			8	1175.28	74.17
			10	2192.52	103.44
			20	5085.36	245.05
			30	9886.81	505.33

Table 6
Sensitivity analysis results for test network 5.

	# of nodes	# of arcs	# of trains	CPU time (s)	% Error in travel time (min)
Agg. 1	18	21	4	166.49	0.00
			8	847.76	0.00
			10	1814.14	1.60
			20	4125.86	7.80
			30	7624.09	10.40
Agg. 2	6	5	4	128.40	0.00
			8	653.40	5.70
			10	1291.99	7.90
			20	2467.47	13.40
			30	4587.00	18.52
Agg. 3	4	3	4	88.68	1.19
			8	246.07	7.24
			10	900.07	13.61
			20	1545.74	16.76
			30	2307.25	17.36

4.2.1. Test network 5

The skeleton structure for test network 5 is shown in Fig. 4.

Each of the aggregate nodes AB, BC, CD and DE are built by randomly assigning it to be either double-track or single-track and then deciding the track length (*L*), track speed-limit (*V*), number of crossings or sidings (*C*) and spacing between crossings or sidings (*S*). The resulting network parameters are shown in Table 4.

AB and BC are double-track networks with 1 and 3 crossings, respectively, distributed uniformly. CD and DE are single-track networks with 3 and 2 sidings, respectively, distributed uniformly. The sidings are 1.5 miles long. If no aggregation is done and each node is set to the length of the maximum train length, then the resulting network model has 31 nodes and 27 arcs. The travel times for this case are shown in Table 5.

AB and BC being double-track networks can be aggregated into a single aggregate node. Similarly, CD and DE can each be aggregated together into a single node. This results in a network with 4 nodes and 3 arcs. The degree of aggregation can also be reduced by having 18 nodes and 21 arcs, and 6 nodes and 5 arcs. The sensitivity analysis test results are summarized in Table 6.

For the aggregate nodes, travel time results are presented as percent error from the no aggregation model (base case). The results show how the percent error in travel time values increase and solution times decrease as the degree of aggregation increases. It takes less time to solve using our recommended solution approach because of the fewer number of nodes and arcs. However, the solution quality is sacrificed. Based on the results for test network 5, we can conclude that “Agg. 2” with 6 nodes and 5 arcs performs almost as well as “Agg. 1” with 18 nodes and 21 arcs. The drop in quality for “Agg. 2” with respect to “Agg. 1” is about 10% when running experiments with 30 trains. In other situations, both perform equally well. However, the solution times to solve the network in the case of “Agg. 2” are less by nearly 60%. Hence, we gain a huge savings in solution time by making some sacrifice in terms of modeling the travel time values.

4.2.2. Test network 6

The skeleton structure for test network 6 is shown in Fig. 5, and the parameters for the aggregate nodes are shown in Table 7.

Without any aggregation, if each node is set to the length of the maximum train length, then the resulting network model has 41 nodes and 57 arcs. Table 8 reports the travel time with varying number of trains.

The results of running sensitivity analysis tests on the above network by varying its degree of aggregation and traffic volume are shown in Table 9. As per its skeleton structure test network 6 can be aggregated at most to 7 nodes and 9 arcs. Due to the structure of the network, it cannot be aggregated any further. .

For this network, “Agg. 1” provides compatible solutions with respect to “No agg.” with significantly smaller CPU run times. Although the run times for “Agg. 2” are faster, the quality of the solution deteriorates significantly at this aggregation level.

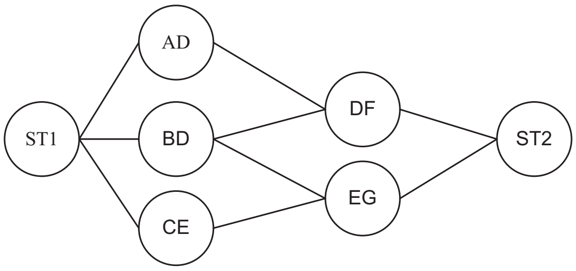


Fig. 5. Skeleton structure for test network 6.

Table 7
Parameters for test network 6.

Node	# of tracks	*L*	*V*	*C*	*S*
AD	Single	15	15	3	1
BD	Single	15	35	4	0.85
CE	Single	15	55	2	1
DF	Double	12.5	55	3	0.75
EG	Double	12.5	35	5	1

Table 8
Travel time for test network 6 without aggregation.

	# of nodes	# of arcs	# of trains	CPU time (s)	Travel time (min)
Test network 6	41	57	4	707.05	104.71
			8	1444.80	117.16
			10	2482.68	141.68
			20	5710.70	172.52
			30	13626.48	209.35

Table 9
Sensitivity analysis results for test network 6.

	# of nodes	# of arcs	# of trains	CPU time (s)	% Error in travel time (min)
Agg. 1	27	36	4	344.48	0.0
			8	959.18	1.2
			10	2101.38	1.9
			20	4145.22	2.4
			30	9487.80	2.8
Agg. 2	7	9	4	131.08	12.0
			8	627.47	19.0
			10	1364.74	26.0
			20	2826.53	32.0
			30	4864.68	40.0

5. Conclusions and future research

In this research, we develop a decision tool that can be used by railway planners to develop good quality routes and initial release schedules, on a daily basis, within a short amount of time, to better manage the limited availability of track capacity. However, our research concentrates only on railway routing and scheduling, without dispatching, for medium to large-scale rail networks.

The model is capable of incorporating route flexibility as opposed to having a train follow a fixed path from its origin to destination. A feature of our modeling strategy is that the planning model ($IP2$) can develop a complete solution, comprising routes and schedules for every train from every node in the network. However, the schedules from the intermediate nodes depend on the operating conditions, congestion levels, train breakdowns, etc. at the time of operation, and in reality often times are determined in real-time by dispatchers. Due to this reason, we focus only on ($IP2$) to determine the routes and release schedules for the trains from their origin stations. As the complexity of ($IP2$) grows with the number of nodes and arcs in the network, it becomes harder to obtain the routes and release schedules in a reasonable amount of time. For this reason, we resort to *aggregation*, wherein we approximate a suitable section of a network as a single node. We use regression-based equations to represent travel time as a function of the network topology, traffic mix and operating parameters. The travel time equations are fed into ($IP2$), which then routes trains along aggregate nodes with the least expected travel time based on the traffic mix (X_i parameters) and the traffic in the opposite direction (D_1 and D_2 parameters). To solve the resulting ($IP2$) for medium and large-scale networks, we develop two approximation-based solution procedures – *R&R_LP* and *R&R_MIP*. Finally, we use a genetic algorithm procedure to search the solution space and improve the solution obtained from the previous two solution procedures. We use 4 test rail networks and an actual rail network from the Los Angeles area to test the performance of the routes and release schedules obtained by our solution procedures. We test the performance by generating a complete solution, including departure schedules from intermediate nodes, using the simulation model developed by Lu et al. (2004). We also compare the quality of the routes obtained from ($IP2$) with the routes used by the greedy heuristic imbedded in the simulation model. Based on the travel times, we conclude that *R&R_MIP_GA* is the best solution procedure. We also conduct sensitivity analyses on two networks to evaluate the quality of the routes and release schedules obtained by *R&R_MIP_GA* to test the effect of the degree of aggregation and traffic volumes on the solution quality.

The model and solution method presented here make some assumptions, including that trains can accelerate and break instantaneously, cannot have part of a train be in two nodes at once, the delay on an aggregate node is a linear regression of the number of passengers. These assumptions and other simplifications were necessary to obtain a mixed integer optimization model that could be solved for real-world sized instances. Future work should look into relaxing some of these assumptions to solve the routing problem for more realistic conditions. Additional work is also needed in benchmarking the proposed approach with exact solution methods or more sophisticated heuristics, perhaps solving for the scheduling of the full network.

References

Boccia, M., Mannino, C., Vasilyev, I., 2013. The dispatching problem on multitrack territories: heuristic approaches based on mixed integer linear programming. Networks 62 (4), 315–326.
Cacchiani, V., Toth, P., 2012. Nominal and robust train timetabling problems. Eur. J. Oper. Res. 219 (3), 727–737.
Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., Wagenaar, K., 2014. An overview of recovery models and algorithms for real-time railway rescheduling. Transport. Res. Part B 63, 15–37.
Cai, X., Goh, C.J., 1994. A fast heuristic for the train scheduling problem. Comput. Oper. Res. 21 (5), 499–510.
Caimi, G., Fuchsberger, M., Laumanns, M., Lüthi, M., 2012. A model predictive control approach for discrete-time rescheduling in complex central railway station areas. Comput. Oper. Res. 39 (11), 2578–2593.
Caprara, A., Fischetti, M., Toth, P., 2002. Modeling and solving the train timetabling problem. Oper. Res. 50 (5), 851–861.
Caprara, A., Kroon, L.G., Monaci, M., Peeters, M., Toth, P., 2006. Passenger railway optimization. In: Barnhart, C., Laporte, G. (Eds.), Handbooks in Operations Research and Management Science, vol. 14, pp. 129–187.
Carey, M., Lockwood, D., 1995. A model, algorithms and strategy for train pathing. J. Oper. Res. Soc. 46, 988–1005.
Cordeau, J., Toth, P., Vigo, D., 1998. A survey of optimization models for train routing and scheduling. Transport. Sci. 32 (4), 380–404.
Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2010. A tabu search algorithm for rerouting trains during rail operations. Transport. Res. Part B 44 (1), 175–192.
Corman, F., D'Ariano, A., Pacciarelli, D., Pranzo, M., 2014. Dispatching and coordination in multi-area railway traffic management. Comput. Oper. Res. 44, 146–160.
D'Ariano, A., Pacciarelli, D., Pranzo, M., 2007. A branch and bound algorithm for scheduling trains in a railway network. Eur. J. Oper. Res. 183 (2), 643–657.
D'Ariano, A., Corman, F., Pacciarelli, D., Pranzo, M., 2008. Reordering and local rerouting strategies to manage train track in real time. Transport. Sci. 42 (4), 405–419.
Dessouky, M.M., Lu, Q., Zhao, J., Leachman, R.C., 2006. An exact solution procedure for determining the optimal dispatching times for complex rail networks. IIE Trans., 141–152
Dorfman, M.J., Medanic, J., 2004. Scheduling trains on a railway network using a discrete event model of railway traffic. Transport. Res. Part B 38, 81–98.
Gen, M., Cheng, R., 2000. Genetic Algorithms and Engineering Optimization. John Wiley & Sons, New York.
Hallowell, S.F., Harker, P.T., 1998. Predicting on-time performance in scheduled railroad operations: methodology and application to train scheduling. Transport. Res. Part A 32 (4), 279–295.
Higgins, A., Kozan, E., Ferreira, L., 1996. Optimal scheduling of trains on a single line track. Transport. Res. Part B 30 (2), 147–161.
Hillestad, R., Van Roo, B.D., Yoho, K.D., 2009. Key Issues in Modernizing the U.S. Freight-Transportation System for Future Economic Growth. Technical Report, RAND Supply Chain Policy Center.
Huntley, C.L., Brown, D.E., Sappington, D.E., Markowicz, B.P., 1995. Freight routing and scheduling at CSX transportation. Interfaces 25 (3), 58–71.

Kecman, P., Corman, F., D'Ariano, A., Goverde, R.M.P., 2013. Rescheduling models for railway traffic management in large-scale networks. Public Transp. 5 (1–2), 95–123.
Kraay, D.R., Harker, P.T., 1995. Real-time scheduling of freight railroads. Transport. Res. Part B 29B (3), 213–229.
Kraay, D.R., Harker, P.T., Chen, B., 1991. Optimal pacing of trains in freight railroads: model formulation and solution. Oper. Res. 39 (1), 82–99.
Lu, Q., Dessouky, M.M., Leachman, R.C., 2004. Modeling of train movements through complex networks. ACM Trans. Model. Comput. Simul. 14, 48–75.
Lusby, R.M., Larsen, J., Ryan, D., Ehrgott, M., 2011a. Routing trains through railway junctions: a new set-packing approach. Transport. Sci. 45 (2), 228–245.
Lusby, R.M., Larsen, J., Ehrgott, M., Ryan, D., 2011b. Railway track allocation: models and methods. OR Spectrum 33 (4), 843–883.
Meng, L., Zhou, X., 2014. Simultaneous train rerouting and rescheduling on an N-track network: a model reformulation with network-based cumulative flow variables. Transport. Res. Part B 67, 208–234.
Mu, S., Dessouky, M.M., 2011. Scheduling freight trains traveling on complex networks. Transport. Res. Part B 45, 1103–1123.
Murali, P., 2010. Strategies for Effective Rail Track Capacity Use. Ph.D. Thesis. University of Southern California.
Murali, P., Dessouky, M.M., Ordóñez, F., Palmer, K., 2010. A delay estimation technique for single and double-track railroads. Transport. Res. Part E 46, 483–495.
Nachtigall, K., Voget, S., 1996. A genetic algorithm approach to periodic railway synchronization. Comput. Oper. Res. 23 (5), 453–463.
Nachtigall, K., Voget, S., 1997. Minimizing waiting times in integrated fixed interval timetables by upgrading railway tracks. Eur. J. Oper. Res. 103, 610–627.
Papadimitriou, C.H., Steiglitz, K., 1998. Combinatorial Optimization. Dover.
Pellegrini, P., Marlière, G., Rodriguez, J., 2014. Optimal train routing and scheduling for managing traffic perturbations in complex junctions. Transport. Res. Part B 59, 58–80.
Pritsker, A.A.B., O'Reilly, J.J., 1999. Simulation with Visual SLAM and AweSim, second ed. John Wiley and Sons, New York and Systems Publishing Corporation, West Lafayette, Indiana.
Şahin, İ., 1999. Railway traffic control and train scheduling based on inter-train conflict management. Transport. Res. Part B 33, 511–534.
Salim, V., Cai, W., 1996. Scheduling cargo trains using genetic algorithms. In: IEEE International Conference on Evolutionary Computation, pp. 224–227.
Salim, V., Cai, W., 1997. A genetic algorithm for railway scheduling with environmental considerations. Environ. Softw. 12 (4), 301–309.
Suteewong, W., 2006. Algorithms for Solving the Train Dispatching Problem for General Networks. Ph.D. Thesis. University of Southern California.
Tornquist, J., Persson, J.A., 2007. N-tracked railway traffic re-scheduling during disturbances. Transport. Res. Part B 41 (3), 342–362.
Zhou, X., Zhong, M., 2007. Single-track train timetabling with guaranteed optimality: branch-and-bound algorithms with enhanced lower bounds. Transport. Res. Part B 41 (3), 320–341.