Stochastic Passenger Train Timetabling using a Branch and Bound Approach

Liang Liu and Maged Dessouky

Department of Industrial and Systems Engineering
University of Southern California, Los Angeles, California, 90089, US

Abstract

We investigate the integrated scheduling of freight and passenger trains in complex railway networks and the timetabling for passenger trains to improve the efficiency of the schedules. In the proposed model, the objective is to study the robust passenger train timetable considering the uncertainty in freight train departure times. The joint routing and scheduling are considered between the two types of trains. We propose a branch and bound framework with hybrid heuristics to solve the problem. On each search tree node, a Lagrangian method is first developed to solve for a relaxed passenger train schedule and then a feasibility recovery heuristic is applied. Then a labeling algorithm is proposed to jointly optimize the freight and passenger trains sequentially. Numerical experiments show the proposed solution approach outperforms other heuristics on actual railway networks.

Keywords: Train timetable, robustness, routing and scheduling

1. Introduction

The train timetable provides a timeline for the operations and activities of the trains over the rail networks. A well-performed timetable is designed based on the passenger demand and the network structure, and needs to be updated when the demand and the network structure changes. In order to provide a reliable service, the railway industry has shown great interest in designing and maintaining a highly efficient train timetable.

The Train Timetabling Problem (TTP) studies the design of the periodic timetable for a set of trains considering the track capacity, the safety headway and other operational constraints. The increasing demand in the railway industry pushes for a better usage of the railway infrastructure and resources, which requires a cooperative planning and scheduling for different kinds of trains, including passenger and freight trains. Timetables serve as the key control variable for the temporal and spatial decisions over the complex networks, and it directly affects the flow times and the resource utilization considering that some rail networks serve both passenger and freight trains. In real world railway operations, train timetables are often affected by unpredictable events, such as unexpected delays, temporary signal failure, track maintenance and equipment breakdown. The actual times of arrival and departure could be significantly different from the planned timetable. The deviation not only creates primary delays, but also leads to secondary delays which further propagates along the route of the trains.

In this paper, we present the stochastic timetabling model for passenger trains. The robustness of the passenger train timetable is important to defend against the stochastic disturbances during its operation. A common approach to improve the robustness of the train schedule is to insert buffer time in the timetable. However, the buffer time usually increases the delay of the trains and decreases the customer experience. In this study, instead of inserting...
buffer time to the passenger train schedule, we are more interested in leveraging the flexible routes to avoid the delays upon disturbances. To plan a robust timetable under the stochastic features in the system, we are particularly interested in dealing with uncertainty from the departure time of the freight trains, since in many networks freight and passenger trains share the same track resource and the departure times of freight train can be random due to the uncertainty in loading and unloading time, etc. We first define the problem and propose a stochastic optimization model in Section 3. Then the overall branch and bound solution framework is presented in Section 4, which includes the master problem solution approach and the subproblem approach. Finally, computational experiments are used to illustrate the performance of the proposed model and solution approach and the results are analyzed in Section 5.

2. Literature Review

The train timetabling problem aims at planning the periodic timetable for the operation of trains that do not violate track capacities and satisfy the operational constraints. Lusby et al. (2011) surveyed the literature on the train timetabling, dispatching, platforming, and routing problems. Harrod (2012) surveyed four timetable formulations suitable for optimization. The survey discussed the models according to their features such as periodic, aperiodic, explicit track, and event only. Cacchiani and Toth (2012) offered a comprehensive survey on the train timetabling problem for its nominal and robust versions. In the nominal version, the cyclic (or periodic) schedule for a daily planning horizon and the non-cyclic schedule in a congested network is studied. In the robust version, the problem of identifying a schedule that avoids disruptions and reduces delay propagation is studied. The trade-off between the nominal objective and the robustness of the timetable is discussed in detail. Caimi et al. (2017) presented an extensive survey on models for railway timetable optimization. The survey puts an emphasis on passenger railway service, and the combinatorial optimization models, solution approaches and applications in practice are discussed in detail. The studied timetabling problems mainly focused on developing either nominal timetabling models for a deterministic environment or robust timetabling models for a dynamic and stochastic environment.

In the nominal timetabling problem, the timetable is designed under deterministic constraints such as demand, delay and travel time. The timetable is planned under track capacity constraints and safety travel constraints, with the objective according to the criteria set by the railway operation company, such as minimizing the waiting time, maximizing passenger satisfaction, or minimizing the total travel time. Either a cyclic or non-cyclic version of the timetable is designed based on the level of planning. Szpieg (1973) formulated the railway timetabling problem as a mixed integer programming model. The routes and departure times of the trains were given and the study focused on the timetabling for a single track railway network. A branch and bound algorithm is proposed to efficiently solve the model. Higgins et al. (1996) focused on studying the train priority in timetabling when conflicts occur at the trains’ meet and pass points. A branch and bound approach is proposed to obtain a near optimal solution efficiently for reasonable size problems. Ghoseiri et al. (2004) developed a multi-objective optimization model for the passenger train timetabling problem in a railway network with single and multiple tracks, as well as multiple platforms with different train capacities. In Carey and Crawford (2007) and Castillo et al. (2011), the scheduling
of the trains in a general network are solved by sequentially optimizing the route and timetable alternatively. Lee and Chen (2009) addressed the scalability of the timetabling problem for real-size instances. A novel optimization heuristic is proposed, which optimizes both the train pathing and timetabling problems. Xu et al. (2014) studied a travel advanced strategy with a combination of an improved TAS algorithm and genetic algorithm, to solve the optimal balanced train timetable with the least delay ratio as well as the optimal train velocity strategy for a single line network. Meng and Zhou (2014) proposed an integer programming model for train dispatching and timetabling on an N-track network. In the formulation, the infrastructure capacity is translated to a vector of cumulative flow variables and then the subproblem is solved as a time-dependent shortest path algorithm using a Lagrangian relaxation framework. Castillo et al. (2016) presented a time partitioning method in solving the railway line planning and timetable optimization problem. Two partition strategies are presented in the study: partitioning based on equal duration and partitioning based on equal number of active trains. Yang et al. (2016) integrated the stop planning and train scheduling problems together in a collaborative optimization model, with the objective function of minimizing the total delay at the origin station and dwelling time at intermediate stations. Robenek et al. (2016) accounted for the passenger satisfaction in the design of the timetable. The model aims at maximizing the train operating company’s profit while maintaining $\epsilon$ level of passenger satisfaction. One limitation of the model is that the conflicts between trains are not considered. Analysis on various values of $\epsilon$ shows that passenger satisfaction can be improved while maintaining a low profit loss for the railway company. Zhou and Teng (2016) studied the passenger train routing and timetabling problem on a rail network consisting of both unidirectional and bidirectional tracks. Borndorfer et al. (2016) investigated a periodic timetabling model with integrated passenger routing. The study shows that different routing models can have a significant influence on the quality of the overall schedule. Guo et al. (2017) studied the train timetable optimization problem for metro transit networks. To enhance the transfer synchronization between lines, timetables are adjusted at transfer stations.

Most nominal timetabling models aim at the efficiency of the railway operation. In comparison, robust timetabling models aim to reduce the impact of random disturbances to avoid delay propagation. One common approach to improve the robustness is to insert slack or buffer time into the timetable. Buffer time absorbs the propagated delay from unexpected disturbances and increases the robustness of the timetable. However, this is in the opposite direction of minimizing the efficiency. Thus a trade-off exists between efficiency and robustness of the timetable. We next present a literature review on robust timetabling models and algorithms.

The majority of the early research on the railway scheduling and timetabling problem assumes perfect information about the system state and focuses on generating the optimized train schedules efficiently. In real-time railway operation, it is common that disturbances arise and the planned timetable becomes infeasible. There are mainly two streams of research to deal with disturbances. One is from a recovery perspective, which is to adjust the routes or reschedule the timetable to recover the feasibility of the schedule. Cacchiani et al. (2014) surveyed the recovery models and algorithms for real-time railway disturbance and disruption management. The survey covers recent mathematical models that algorithmically solve railway rescheduling problems that are related to the timetable, the rolling stock and crew planning. The train timetable rescheduling problem aims at the adjustment of an existing timetable that has become infeasible due to unpredicted disturbances or disruptions. The common adjustment decisions considered in the literature include the routing of trains, the time instance of the departures and arrivals.
of the trains and the order of the trains on their common track sections. Another stream of research is to design a robust model to determine the timetable. Railway real time traffic management aims to minimize delays and tardiness when unpredicted disturbances of operations occur. In stochastic timetabling models, the objective is to find the best train routing and scheduling in case of disturbances. Parbo et al. (2016) gave an extensive overview of the timetable design problem.

The definition of timetable robustness varies in the studies. In the early research, timetable robustness focused on the recovery ability of the timetable to absorb disruptions, as in Vromans (2005), Salido et al. (2008), Cacchiani et al. (2009) and Medeossi et al. (2009). In recent research, the robustness definition considers more about the delay absorption and travel time uncertainty reduction by maintaining the trade-off between nominal timetable and robust timetable, see Schbel and Kratz (2009) and Dewilde et al. (2011). Following the idea of robust optimization, timetable robustness can be improved by introducing buffer times that are large enough to absorb the most frequently occurring delays. Generally, robust timetables may add too much buffer times which makes the plan conservative. Regarding this issue, a common approach is to bound the total amount of buffer times and decide how to optimize the allocation. The approach follows the idea of light robustness introduced by Fischetti and Monaci (2009) to bound the maximum deterioration of the cost function. Meng and Zhou (2011) developed a stochastic programming model and a rolling horizon decision process for train scheduling on a single-track rail line. The model periodically optimizes the dispatching schedule for a long rolling horizon, while determining a robust meet-pass plan for every rolling period. Pellegrini et al. (2014) proposed a mixed integer programming model to find the best train routing and schedule in case of perturbation in the real time railway traffic management problem. The MILP formulation is tested on two types of instances, random instances and perturbations of real instances. The experiments show that the granularity of the representation of the control area has a significant impact on the solution quality, which is consistent with the study of Corman et al. (2009). Sun et al. (2014) focused on timetable design for metro services under dynamic temporal passenger demand. Three optimization models are proposed to design the demand-sensitive timetable. Dewilde et al. (2014) introduced an iterative approach to successively improve the routing of trains by alternatively updating the timetable using tabu search. The scope of the problem is limited to the station area. The routing module optimizes the total weighted timespan for a given timetable. Then with the same objective function, the timetable is adjusted using discretized time units of 0.1 minutes. Besinovic et al. (2015) provided a hierarchical framework for timetable design using microscopic and macroscopic models of the network. An integrated approach for computing a microscopically conflict-free timetable is presented. Sels et al. (2016) proposed an optimization approach for determining the passenger timetable, which consists of two step actions called reflowing and retiming. Hassamayebi et al. (2016) studied applying robust stochastic programming models to the train timetabling problem for urban rail transit systems. The dynamics and uncertain demand is represented by scenario-based arrival rates of passengers. Burggraeve et al. (2017) studied the integrated railway line planning and timetabling model, and proposed a heuristic algorithm to interactively optimize the two objectives. The line planning model aims at minimizing the operator and passenger costs, while the timetabling model aims at maximizing the buffer times between any pair of trains. In some other recent studies, Tornquist and Persson (2007), D’Ariano et al. (2007), Corman et al. (2010), Fu and Dessouky (2017) studied various topics of the train scheduling problem, including timetabling in a general N-track network and real time
scheduling to minimize the delay propagation under various sources of disturbances.

It is noted that most of the robustness-oriented timetable studies concentrated on the buffer time allocation. For a single line railway system, allocating buffer time is the most effective way to absorb the disturbance and delay, and also to maintain the safety headway between trains. However, for a general railway system with multiple tracks, more choices are available by introducing flexible routing. The disturbances create primary delay for trains, but this delay can be minimized by rerouting the train to other feasible routes. The complexity of the problem significantly increases when routing decisions are introduced into the system.

3. Model Formulation

In the railway market, freight trains usually have a more flexible schedule than passenger trains. Due to the uncertainties in the freight loading, cars sorting and assembling, refueling and crew changes, the departure times can have significant variability. Upon the different freight train departure times, their schedules, routes and the precedence relationship with other trains could vary accordingly. We study the timetable of passenger trains considering the impact of the uncertainty in the freight train departure times.

The schedule of the passenger trains is based on the passengers' demand, which is quantified by the number of passengers traveling from station to station. Instead of concentrating on the deviation time from the timetable, we consider the sum of the passenger travel times, in which the travel time between two stations is weighted by the number of passengers on the train. The passenger demand, which can be approximated by averaging the daily historical data, is assumed to be deterministic. The interaction and conflict for railway network resources between passenger trains and freight trains are also included in the model. The conflict happens when the trains compete for the same track resource at the same time. The time of arrival at the node is determined by two factors, the departure time at the origin station and the travel time along the route. As previously mentioned, the freight train departure time is an uncertain factor, and the actual departure time of freight trains significantly affects the passenger trains' timetable.

To take the uncertainty in freight train departure times into account, we formulate a two-stage optimization problem. The objectives of the two stage problems are different. The first stage problem aims to optimize the passenger trains' operation, while the second stage is to adjust the passenger trains operation based on the freight trains' actual departure time.

The notations of the model are presented as follows.

\[
\begin{align*}
Q_f & \quad \text{Set of freight trains} \\
Q_p & \quad \text{Set of passenger trains} \\
N & \quad \text{Node set of network} \\
O_q & \quad \text{The origin node of train } q\text{'s route} \\
D_q & \quad \text{The destination node of train } q\text{'s route} \\
S_q & \quad \text{The set of station stop nodes along passenger train } q\text{'s route, including origin and destination} \\
N_{q, ED}^e & \quad \text{The auxiliary dummy end node after destination } D_q \\
\end{align*}
\]
\[ N^+_q \] The subset of nodes which are reachable for train \( q \)
\[ N^-_i,q \] The preceding set of nodes for train \( q \) before entering node \( i \)
\[ N^+_i,q \] The succeeding set of nodes for train \( q \) after exiting node \( i \)
\[ T^e \] The end time of daily operation
\[ \mu \] Minimum safety headway between trains
\[ M \] A sufficiently large number
\[ T^d_q \] The earliest departure time for passenger train \( q \) from its origin station
\[ p_{q,i,j} \] The number of passengers traveling from station node \( i \) to station node \( j \) on train \( q \)
\[ V \] Total number of scenarios, details discussed in Section 4.1
\[ \omega^v_q \] The departure time of freight train \( q \) from its origin station in scenario \( v \in V, \forall q \in Q_f \)
\[ \beta \] The weight of total deviations time from passenger train timetable in the subproblem objective function

### First stage decision variables:

\[ t^a_{q,i} \] The arrival time of passenger train \( q \) to node \( i \), \( \forall q \in Q_p, i \in N^+_q \)
\[ t^d_{q,i} \] The departure time of passenger train \( q \) from node \( i \), \( \forall q \in Q_p, i \in N^+_q \)
\[ I_{q,i,j} \] Binary routing variable to indicate if passenger train \( q \) travels from node \( i \) to its successor node \( j \), \( \forall q \in Q_p, i \in N^+_q, j \in N^+_i,q \)
\[ x_{q_1,q_2,i} \] Binary precedence variable in master problem to indicate if passenger train \( q_1 \) passes node \( i \) before passenger train \( q_2 \), \( \forall q_1, q_2 \in Q_p, q_1 \neq q_2, i \in N^+_q \cap N^+_q \)

### Second stage decision variables:

\[ \hat{t}^a_{q,i} \] The adjusted arrival time of train \( q \) to node \( i \) if \( q \in Q_p \), the computed arrival time of train \( q \) to node \( i \) if \( q \in Q_f \), \( i \in N^+_q \)
\[ \hat{t}^d_{q,i} \] The adjusted departure time of train \( q \) to node \( i \) if \( q \in Q_p \), the computed departure time of train \( q \) to node \( i \) if \( q \in Q_f \), \( i \in N^+_q \)
\[ \hat{I}_{q,i,j} \] Binary routing variables, \( \forall q \in Q_f, i \in N^+_q, j \in N^+_i,q \). Note that \( \hat{I}_{q,i,j} = I_{q,i,j}, \forall q \in Q_p \)
\[ \hat{x}_{q_1,q_2,i} \] The adjusted precedence variables if \( q_1, q_2 \in Q_p, q_1 \neq q_2 \), the computed precedence variables if \( q_1, q_2 \in Q_p, q_1 \neq q_2 \) or \( q_1 \in Q_f, q_2 \in Q_p \) or \( q_1 \in Q_p, q_2 \in Q_f \) or \( q_1, q_2 \in Q_f, q_1 \neq q_2 \), \( i \in N^+_q \cap N^+_q \)

In the first stage problem, the passenger trains’ routes and schedules are optimized based on the passenger demand. The passenger train time decisions, including the arrival times \( t^a_{q,i} \), the departure times \( t^d_{q,i} \), the routing decisions \( I_{q,i,j} \) and the precedence decisions \( x_{q_1,q_2,i} \) are defined as the first-stage decisions \((t, I, x)\). We define the first stage decision problem as the master problem. In the master problem, the objective of the optimization is to minimize the sum of the passengers travel time and the expectation of the objective value of the subproblems.

In the second stage problem, the freight trains’ schedules are optimized based on the departure times, and the
passenger trains’ schedules are adjusted accordingly. At the beginning of the second-stage freight train scheduling, a realization of the actual departure times of the freight trains unfolds, and is denoted as \( \omega^v_q, q \in Q_f \), where \( v \) is the index for scenario. A scenario represents an instance of the combination of the freight trains departure times at their origin stations. In a scenario, the first stage decision needs to be adjusted to make a reactive decision \((\hat{t}, \hat{I}, \hat{x})\), which also includes the decision variables for the freight trains in the scenario. We define the second-stage problem as a subproblem. In the subproblem, the objective is to minimize the weighted sum between the freight trains total travel time and the deviations time from the passenger trains timetable.

In the master problem, the objective function contains the total passenger train travel time weighted by the number of passengers between two stations and the expectation of the subproblem objective function. The expectation term evaluates the average objective values of the subproblems of all the scenarios. In the subproblem, the objective function contains the freight trains total travel time and the deviations of the passenger train schedule from the timetable. These two components in the subproblem objective function are weighted by a parameter \( \beta \), which reflects the priority between freight and passenger trains in the integrated schedule. We will investigate the choice of \( \beta \) and its impact on the derived schedules.

The constraints space contains the travel time constraints, track capacity constraints, safety headway constraints and domain constraints. Additionally, each passenger train is assigned with an earliest departure time \( T_{d_q} \) from its origin station, and the combination of freight train departure times from origin station is denoted as \( \omega^v_q \) for scenario \( v \).

Next, we formally formulate the two stage mixed integer optimization model, including the master problem \( Q_m \) and the subproblem \( Q_s \). The master problem solution of the passenger train arrival/departure times decision variables \( t \) and routing decision variables \( I \) are shared with the subproblems through the expectation function. The master problem \( Q_m \) is formulated as follows.

\[
\min \sum_{q \in Q_p} \sum_{i,j \in S_q} p_{q,i,j} (t^a_{q,j} - t^a_{q,i}) + E_{\omega}[h(t, I, \omega)]
\]

Subject to:

\[
\sum_{j \in N_{q,q}^+} I_{q,j} = 1, \quad \forall q \in Q_p \tag{2}
\]

\[
\sum_{i \in N_{q,q}^-} I_{q,i} = 1, \quad \forall q \in Q_p \tag{3}
\]

\[
\sum_{j \in N_{q,q}^+} I_{q,j,s} = 1, \quad \forall q \in Q_p, s \in S_q \setminus \{Q_q, D_q\} \tag{4}
\]

\[
\sum_{j \in N_{q,q}^+} I_{q,j} = \sum_{k \in N_{j,q}^+} I_{q,j,k}, \quad \forall q \in Q_p, \forall j \in N_{q,q}^+ \tag{5}
\]

\[
(1 - I_{q,i,j})M^a + t^a_{q,j} - t^a_{q,i} \geq B_1^1, \quad \forall q \in Q_p, i \in N_{q,q}^+, j \in N_{i,q}^+ \tag{6}
\]

\[
t^d_{q,D_q} - t^d_{q,D_q} \geq B_2^1, \quad \forall q \in Q_p \tag{7}
\]

\[
(1 - I_{q,i,j})M^a + t^d_{q,j} - t^d_{q,i} \geq B_1^1, \quad \forall q \in Q_p, i \in N_{q,q}^+, j \in N_{i,q}^+ \tag{8}
\]

\[
(1 - I_{q,i,j})M^a + t^d_{q,i} - t^d_{q,j} \geq B_1^1 - B_1^1, \quad \forall q \in Q_p, i \in N_{q,q}^+, j \in N_{i,q}^+ \tag{9}
\]
\begin{align*}
&(1 - x_{q_1,q_2,i})M + t_{q_2,i}^a \geq t_{q_1,i}^d + \mu, \ \forall q_1, q_2 \in Q, i \in N_{q_1}^t \cap N_{q_2}^t \tag{10} \\
&(2 - \sum_{j \in N_{q_1}^+} I_{q_1,i,j} - \sum_{k \in N_{q_2}^+} I_{q_2,i,k})M + x_{q_1,q_2,i} M + t_{q_1,i}^a \geq t_{q_2,i}^d + \mu, \ \forall q_1, q_2 \in Q, i \in N_{q_1}^t \cap N_{q_2}^t \tag{11} \\
&x_{q_1,q_2,i} \leq \sum_{j \in N_{q_1}^t} I_{q_1,i,j} + \sum_{k \in N_{q_2}^t} I_{q_2,i,k}, \ \forall q_1, q_2 \in Q, i \in N_{q_1}^t \cap N_{q_2}^t \tag{12} \\
&t_{q,D_q}^a \leq T^c, \ \forall q \in Q_p \tag{13} \\
&t_{q,O_q}^a \geq T_q^d, \ \forall q \in Q_p, i \in N_q^t \tag{14} \\
&t_{q,i}^d \geq t_{q,i}^a, \ \forall q \in Q_p, i \in N_q^t \tag{15} \\
&I_{q,i,j} \in \{0,1\}, \ \forall q \in Q_p, i,j \in N_q^t \tag{16} \\
x_{q_1,q_2,i} \in \{0,1\}, \ \forall q \in Q, i \in N_{q_1}^t \cap N_{q_2}^t \tag{17}
\end{align*}

The objective function \[\text{(1)}\] minimizes the total weighted travel time of the passenger trains and the expectation of the subproblem objective. \(h(\mathbf{t}, \mathbf{I}, \omega)\) denotes the optimal objective value of the subproblem, given the master problem decisions and the realization of the freight train departure times for each scenario as the input. Constraints \(\text{(2)} - \text{(3)}\) ensure the route of a train has to start from the origin node and end at the destination node. Constraint \(\text{(4)}\) states that passenger trains have to visit their intermediate station stops. Constraint \(\text{(5)}\) guarantees the flow conservation on each node. Constraints \(\text{(6)} - \text{(8)}\) ensure the minimum travel time on each node. If the train encounters any waiting such as congestion, the travel time is greater than the minimum travel time which is the free flow travel time. Constraint \(\text{(9)}\) ensures the minimum travel time for a train to completely clear the occupation of the current node. Constraints \(\text{(10)} - \text{(11)}\) are the deadlock avoidance mechanism that keeps the distance between the trains to be above the minimum safety headway. Constraint \(\text{(12)}\) forces \(x_{q_1,q_2,i} = 0\) when both trains \(q_1\) and \(q_2\) do not travel on node \(i\). Constraint \(\text{(13)}\) states that the departure time of a passenger train from the origin station cannot be earlier than the scheduled departure time. In Constraint \(\text{(14)}\), the earliest departure time of a passenger train \(q\) can not be earlier than the earliest departure time \(T_q^d\).

In the subproblem \(Q^v\), the freight train schedules are optimized and the passenger train schedules are adjusted in the realization of the freight train departure times \(\omega\). We index the scenario with \(v\) for a realization of the freight trains departure times \(\omega^v\). The subproblem \(Q^v\) is also indexed by \(v\) and is formulated as follows:

\[h(\mathbf{t}, \mathbf{I}, \omega^v) = \min \sum_{q \in Q_f} (\tilde{t}_{q,D_q}^d - \tilde{t}_{q,O_q}^d) + \beta \sum_{q \in Q_p} \sum_{i,j \in S_q} (\tilde{t}_{q,j}^a - t_{q,j}^a) \tag{18}\]

Subject to:

\[\sum_{j \in N_{q}^+} \tilde{I}_{q,O_q,j} = 1, \ \forall q \in Q_f \tag{19}\]

\[\sum_{i \in N_{q}^+} \tilde{I}_{q,i,D_q} = 1, \ \forall q \in Q_f \tag{20}\]

\[\sum_{j \in N_{q}^+} \tilde{I}_{q,i,j} = \sum_{k \in N_{q}^+} \tilde{I}_{q,i,k}, \ \forall q \in Q_f, \forall j \in N_{q}^t \tag{21}\]
\( (1 - \tilde{I}_{q,i,j})M + \tilde{t}_{q,i,j}^a - \tilde{t}_{q,i}^a \geq B^1_{q,i} \) \( \forall q \in Q_p \cup Q_f, i \in N^t_q, j \in N^t_{i,q} \)

\( (1 - \tilde{I}_{q,i,j})M + \tilde{t}_{q,i,j}^d - \tilde{t}_{q,i}^d \geq B^1_{q,i} \) \( \forall q \in Q_p \cup Q_f, i \in N^t_q, j \in N^t_{i,q} \)

\( (1 - \tilde{I}_{q,i,j})M + \tilde{t}_{q,i,j}^d - \tilde{t}_{q,i}^d \geq B^2_{q,i,j} \) \( \forall q \in Q_p, i \in N^t_q, j \in N^t_{i,q} \)

\( \tilde{t}_{q,O_q} = \omega^a_q \) \( \forall q \in Q_f \)

\( \tilde{t}_{q,D_q} \leq T^e_q \) \( \forall q \in Q_p \)

\( \tilde{I}_{q,i,j} = I_{q,i,j} \) \( \forall q \in Q_p, i \in N^t_q, j \in N^t_{i,q} \)

\( \tilde{x}_{q_1,q_2,i} \in \{0,1\} \) \( \forall q \in Q_f \), \( i \in N^t_{q_1} \cap N^t_{q_2} \)

The subproblem objective function contains two components: one is the total freight train travel time and the other is the deviation from the timetable of the passenger trains in the master problem decisions. Besides the train scheduling constraints which are similar to the master problem, in Constraint (31), the actual departure times of the freight trains are fixed as the realization of the departure time \( \omega^a_q \), in which \( v \) is the scenario index corresponding to the underlying subproblem. This constraint enforces the departure times, thus partially limiting the freedom in the freight train schedule. In the solution approach we propose, the overall two-stage problem is decomposed, and the master problem and subproblem solution procedures are applied iteratively.

4. Overall Solution Framework

In the proposed two-stage stochastic optimization model, the master problem and subproblems are binded by the first stage decision variables, including the passenger trains’ routing decisions. Once the master problem decisions are made and the scenario in the subproblem is realized, further decisions are needed to finalize the scenario specific to the freight trains related decisions, and to adjust the schedules of the passenger trains. Note that the objective values of the subproblems are fed back to the master problem and influences the master problem decision. Thus, a systematic mechanism is required to integrate the optimization of the master problem and subproblems.
To solve this large scale mixed integer stochastic model, a hybrid heuristic algorithm under a branch and bound framework is developed. The joint routing and scheduling scheme is extended to systemically generate branching nodes to reduce the conflicts between passenger and freight trains. Each branching node, which has an enhanced problem of its parent node, goes through the master problem solution procedure (Section 4.1) and subproblem solution procedure (Section 4.2). Then further branching is applied as in Section 4.3. To reduce the branching search space complexity, a bounding rule is embedded into the branch and bound framework as discussed in Section 4.3.3. In Figure 1, we present the overall solution framework. In the solution framework, a *search tree node* includes a combination of the master problem and subproblems. The algorithm explores the search tree, solves the problems on each search tree node, and finds the best train routes and schedule.

First we introduce the notation to be used in the branch and bound framework.
The main steps of the branch and bound algorithm in Figure 1 are summarized as follows.

**Step 1: Initialization**

Initialize the network, passenger demand and schedule data. Let $ANL = \emptyset$, $k^* = 0$. Construct the root node $c(0) = \{Q^0_m, \bigcup_{v=1}^V Q^v_s\}$, and push $c(0)$ to the active node list $ANL$. $Q^0_m$ is constructed using formulation (1) - (17). $Q^v_s$ is the subproblem for scenario $v$.

**Step 2: Node Selection**

Select a node $c(k)$ from $ANL$ using breadth first search (BFS).

**Step 3: Master Problem $Q^k_m$ Solution Procedure**

**Substep 3.1: Relaxed Passenger Schedule**

Solve a relaxed version of $Q^k_m$ to get a relaxed precedence decisions, which has feasible time and routing decisions but relaxed precedence decisions.

**Substep 3.2: Passenger Train Feasibility Recovery**

Run the feasibility recovery algorithm in Section 4.1.2 to recover the feasibility from the relaxed solution in Step 3.1, and get a feasible master problem solution $\{t, I, x\}$ of $Q^k_m$.

**Step 4: Subproblem $Q^v_s$ Solution Procedure**

Run the labeling algorithm in Section 4.2.1 to retrieve the deadlock free freight train schedules in $Q^v_s$, and adjust the passenger train schedules accordingly. Note that the solution procedures for each subproblem $Q^v_s$ are independent. Thus, the solution approaches in this step are performed in parallel.

**Step 5: Objective Estimation**

Apply the sample average on the objective value of $\bigcup_{v=1}^V Q^v_s$ with the corresponding adjusted master problem objective, and update $obj(k)$.

**Step 6: Bounding**

Check if the bounding rule discussed in Section 4.3.3 is met. If not met, update $k^* = k$ if $obj(k) < obj(k^*)$ and continue.

**Step 7: Stopping Criteria Checking**

Stop the search if any one of the following criteria is met: 1. $ANL$ is empty; 2. Maximum computation time is reached; 3. Computer memory limit is reached.

**Step 8: Congestion Evaluation**
Evaluate the conflict of each passenger train in the solution of \(c(k)\) as discussed in Section 4.3.1.

**Step 9: Branching**

Apply the branching strategy in Section 4.3.2 to generate the branching leaf nodes \(\{c(k+1), c(k+2), \ldots\}\), and append the leaf nodes to ANL. Then remove the current node \(c(k)\) out of ANL.

**Step 10: Output Solution**

Output the passenger train routes and timetables from the search tree node \(c(k^*)\) that has the best solution.

We present the detailed solution algorithm to solve \(c(k)\), which contains the solution procedure to the master problem \(Q_m^k\) (see Section 4.1) and the solution procedure to a set of subproblems \(\bigcup_{v=1}^{T} Q_s^v\) (see Section 4.2). The solution to the master problem \(Q_m^k\), which is the feasible schedule of passenger trains, is passed into each of the subproblems. Based on the scenario of the freight train departure times, each of the subproblems \(Q_s^v\) is solved independently. The passenger train schedules are adjusted accordingly in each of the subproblem solution approaches. Finally, the objective value is approximated by averaging the subproblems objective values.

### 4.1. Master Problem Solution Procedure

The master problem \(Q_m^k\) defined in a search tree node is the scheduling problem of only the passenger trains. We assume that in the solution approach of this first stage problem, the passenger trains are scheduled without considering the freight trains. The objective of the master problem is defined as the sum of the weighted passenger traveling time, as the first component of Equation (1). In the objective function, the passenger train travel times between stations are weighted by the number of passengers. Given the passenger demand as the input, the route decisions, time decisions and precedence decisions are optimized under Constraints (2) - (17) and the branching constraint (72) which is discussed in Section 4.3.2.

Due to the complexity of the problem, we are interested in an approximate optimal solution to achieve a trade-off with the solving time. Solving the master problem optimally is computationally prohibitive since an optimal solution algorithm will require significant computation resources and the master problem needs to be solved many times.

The strategy to solve the master problem is to first optimize the relaxed problem to obtain a relaxed passenger trains schedule, and then recover the feasibility from the relaxed solution. Specifically, the relaxed schedule is feasible for the routing decision variables and time decision variables, but are not necessarily guaranteed to be feasible for the precedence decision variables relation constraints. When conflicts between precedence decisions are detected, a feasibility recovery algorithm is employed to resolve the infeasibilities.

#### 4.1.1. Relaxed Passenger Train Scheduling with Dual Ascent Algorithm

The constraints in the master problem can be classified into three sets. The first set of Constraints (2) - (9) and (72) are related to the routing and safety headway of the individual trains. Constraint (72) are the routing constraints to be discussed in Section 4.3.2. The second set of Constraints (10) and (12) define the precedence relationship between the trains at each node. Due to the exponential combination between trains at each node, the number of integer variables and the number of constrains in the second group are large, and it makes the problem hard to optimally solve. The third set of Constraints (13) - (17) are the domain constraints.
To solve the master problem efficiently, we propose a dual ascent algorithm to iteratively optimize the relaxed problem, which finally converges to the optimal solution of the Lagrangian dual. Some of the constraints in the master problem are relaxed by applying the Lagrangian relaxation to the constraints. We apply the dual ascent algorithm using a parallel optimization approach. In the parallel approach, the Lagrangian multipliers are updated iteratively until the convergence criterion is met. The decisions variables in the primal space are separable, so a decomposition is applied and the primal problem is solved efficiently.

We formulate the Lagrangian function by introducing three sets of Lagrangian multipliers \( \pi_{q_1,q_2,i} \), \( \rho_{q_1,q_2,i} \) and \( \gamma_{q_1,q_2,i} \) associated with Constraints (10), (11) and (12), and add them to the objective function to the relaxed problem.

\[
L[(\pi, \rho, \gamma), (x, I, t)] = 
\sum_{q \in Q_p} \sum_{i,j \in S_q} p_{q,i,j} (f_{q,j}^a - f_{q,i}^a) - 
\sum_{q_1,q_2 \in Q_p} \sum_{i \in N_{q_1}^+ \cap N_{q_2}^+} \pi_{q_1,q_2,i} \left( 1 - x_{q_1,q_2,i} \right) M + t_{q_2,i}^a - t_{q_1,i}^a - \mu - 
\sum_{q_1,q_2 \in Q_p} \sum_{i \in N_{q_1}^+ \cap N_{q_2}^+} \rho_{q_1,q_2,i} \left( 2 - \sum_{j \in N_{q_1}^+} I_{q_1,i,j} - \sum_{k \in N_{q_2}^+} I_{q_2,i,k} \right) M + x_{q_1,q_2,i} M + t_{q_1,i}^a - t_{q_2,i}^a - \mu - 
\sum_{q_1,q_2 \in Q_p} \sum_{i \in N_{q_1}^+ \cap N_{q_2}^+} \gamma_{q_1,q_2,i} \left( \sum_{j \in N_{q_1}^+} I_{q_1,i,j} + \sum_{k \in N_{q_2}^+} I_{q_2,i,k} - x_{q_1,q_2,i} \right)
\]

Subject to: Constraints (2) - (9), (13) - (17) in Section 3 and (72) in Section 4.3.2

The Lagrangian dual function is

\[
g(\pi, \rho, \gamma) = \inf_{x, I, t} L[(\pi, \rho, \gamma), (x, I, t)]
\]

By relaxing Constraints (10), (11) and (12), the precedence constraints between passenger trains are relaxed. The Lagrangian dual serves as the lower bound of the primal master problem. To obtain the tightest bound, the following optimization problem is solved:

\[
g^*(\pi, \rho, \gamma) = \max_{\pi, \rho, \gamma \geq 0} g(\pi, \rho, \gamma)
\]

To solve \( g^*(\pi, \rho, \gamma) \), we propose to employ a dual ascent algorithm. Given the dual multipliers \( (\pi, \rho, \gamma) \) and assuming that \( (x, I, t) \) minimizes \( L[(\pi, \rho, \gamma), (x, I, t)] \), the subgradient can be calculated as \( \partial g(\pi, \rho, \gamma) \). A sequence of \( \alpha_1, \alpha_2, ..., \alpha_r \) is selected as the step size. Then the subgradient at iteration \( r \) can be defined as follows:

\[
(x, I, t)^{r+1} = \arg \min_{x, I, t} L[(\pi, \rho, \gamma)^r, (x, I, t)]
\]

\[
(\pi, \rho, \gamma)^{r+1} = (\pi, \rho, \gamma)^r + \alpha_r \partial g(\pi, \rho, \gamma)^r
\]

The convergence rate and effectiveness of the dual ascent algorithm requires a small number of dual constraints. In the Lagrangian dual, the size of the dual constraints is in the space complexity of \( O(N \cdot |Q|) \). However, in
the branching procedure to be discussed later in Section 4.3.2, the extra routing Constraint (72) is added to the master problem within the branching process. Note that the dual multipliers $\pi_{q_1,q_2,i}$, $\rho_{q_1,q_2,i}$ and $\gamma_{q_1,q_2,i}$ are defined on the shared nodes of passenger trains $q_1$ and $q_2$ routes. Constraint (72) forbids the travel of a train to a specific node, and eliminates a subset of dual multipliers with the size of $O(|Q_p|)$. The size of the dual multipliers keeps decreasing, and the size of the dual constraint set shrinks quickly.

We use the following equation to calculate $\alpha_r$, where $\hat{g}$ denotes the objective value of the best solution to $g(\pi, \rho, \gamma)$ found so far. $\hat{g}$ is chosen by solving the master problem with the passenger train scheduling algorithm that was proposed in Liu and Dessouky(2017). The denominator is the two-norm of the dual multiplier vector. The denominator scales down the step size if the norm of the subgradient is too large, which can reduce the oscillation.

$$\alpha_r = \frac{1}{r} \frac{\hat{g} - g(\pi, \rho, \gamma)^r}{\| (\pi, \rho, \gamma)^r \|^2}$$  (42)

To solve the subgradient in Equation (40), we show that it is a separable problem, and the separated subproblems can be solved efficiently. By reformulating Equation (40), the problem can be decomposed as follows.

$$\min_{\mathbf{x}, \mathbf{I}, \mathbf{t}} L[(\pi, \rho, \gamma)^r, (\mathbf{x}, \mathbf{I}, \mathbf{t})] = \min_{\mathbf{I}, \mathbf{t}} \sum_{q \in Q_p} g_q(\mathbf{I}, \mathbf{t}) - \min_{\mathbf{x}} f(\mathbf{x}) + C$$  (43)

In Equation (43), the objective function is decomposed into three components. $g_q(\mathbf{I}, \mathbf{t})$ is the individual train routing and scheduling term, which is defined on each passenger train $q$. $f(\mathbf{x})$ is the precedence relationship term that contains the precedence decisions $\mathbf{x}$ of all the passenger trains. $C$ is a constant term of $(\pi, \rho, \gamma)$, and is constant with respect to $(\mathbf{x}, \mathbf{I}, \mathbf{t})$. Next, we show that the constraint space is also separable.

**Individual train routing and scheduling problem:**

$$\min_{\mathbf{I}, \mathbf{t}} g_q(\mathbf{I}, \mathbf{t})$$

$$= \min_{\mathbf{I}, \mathbf{t}} \sum_{i,j \in S_q} p_{q,i,j}(t^a_{q,i,j} - t^a_{q,i}) - \sum_{q' \in Q_p} \sum_{i : \mathbf{I} \cap N_{q'}^i} (\pi_{q',q,i} + \rho_{q',q,i})t^a_{q,i} - (\pi_{q,q',i} + \rho_{q,q',i})t^d_{q,i}$$

$$- \sum_{q' \in Q_p} \sum_{i : \mathbf{I} \cap N_{q'}^i} \left[ \gamma_{q,q',i} + (\pi_{q,q',i} + \rho_{q,q',i})M \right] \sum_{j \in N_{q,i}^+} I_{q,i,j}$$  (44)

Subject to:

Constraints (2) - (9), (13) - (17) and

$$I_{q,i,j} = 0, i \in \hat{N}_q, j \in \hat{N}_{q,i}^+$$  (45)

In Constraint (45), we define the *forbidden set* as $\hat{N}_q$ as the set of nodes on which the routing variables of passenger train $q$ that are fixed to zero, and define $\hat{N}_{q,i}^+$ as the succeeding node set of train $q$ from node $i$. The forbidden set is formally discussed in Section 4.3.2. Each subproblem is solved using a CPLEX solver.

The decomposition isolates the precedence decisions $\mathbf{x}$ from the individual train routing and scheduling problem. Each subproblem $g_q(\mathbf{I}, \mathbf{t})$ only has the decision variables related to the passenger train $q$. The subproblems are
independent with each other, and thus the solver can be called in parallel to solve each of the subproblems.

**Precedence assignment problem:**

$$\min_{\mathbf{x}} \ f(\mathbf{x})$$

$$= \min_{q_1, q_2 \in Q_p} \ \sum_{i \in N_{q_1}^t \cap N_{q_2}^t} \ (-M \rho_{q_1, q_2, i} + M \pi_{q_1, q_2, i} + \gamma_{q_1, q_2, i}) x_{q_1, q_2, i}$$ \hspace{1cm} (46)

Subject to:

$$x_{q_1, q_2, i} \in \{0, 1\}, \ \forall q \in Q_p, i \in N_{q_1}^t \cap N_{q_2}^t$$ \hspace{1cm} (47)

The precedence decisions of all the passenger trains are included in this problem. The precedence decision variables $$x_{q_1, q_2, i}$$ are unconstrained within its binary integer domain. The optimal solution to this problem can be found by checking the coefficient of each variable in the objective function.

$$x_{q_1, q_2, i} = \begin{cases} 0, & \text{if } -M \rho_{q_1, q_2, i} + M \pi_{q_1, q_2, i} + \gamma_{q_1, q_2, i} \geq 0 \\ 1, & \text{otherwise} \end{cases}$$ \hspace{1cm} (48)

**Constant term:**

$$C = \sum_{q_1, q_2 \in Q_p} \ \sum_{i \in N_{q_1}^t \cap N_{q_2}^t} \mu(\rho_{q_1, q_2, i} + \pi_{q_1, q_2, i}) - M(2\rho_{q_1, q_2, i} + \pi_{q_1, q_2, i})$$ \hspace{1cm} (49)

Note that in Equation (40), the dual multipliers are input as parameters and the problem is optimized for the primal decision variables. The constant term is an expression of the dual multipliers, and is independent of the primal decision variables. In the optimization of Equation (43), it does not change the ascent direction, and thus will not be considered when calculating the subgradient. However the value of $$C$$ is needed to get the optimal objective value of Equation (43) when substituting back into Equation (42) for the calculation of the step size.

By solving the decomposed individual train routing and scheduling problem and the precedence assignment problem, the optimal solution of the primal decision variables ($$\mathbf{x}, \mathbf{I}, \mathbf{t}$$) to the Lagrangian can be found.

With the solution of the primal variables, the subgradient of the dual variables can be calculated by substituting the primal decision variables to the following equations.

$$\frac{\partial L}{\partial \pi_{q_1, q_2, i}} = -t^a_{q_2, i} + t^d_{q_1, i} + M x_{q_1, q_2, i} + \mu - M$$ \hspace{1cm} (50)

$$\frac{\partial L}{\partial \rho_{q_1, q_2, i}} = M(\sum_{j \in N_{q_1}^+} I_{q_1, i, j} + \sum_{k \in N_{q_2}^+} I_{q_2, i, k}) - t^a_{q_2, i} + t^d_{q_2, i} + \mu - M x_{q_1, q_2, i} - 2M$$ \hspace{1cm} (51)

$$\frac{\partial L}{\partial \gamma_{q_1, q_2, i}} = -\sum_{j \in N_{q_1}^+} I_{q_1, i, j} - \sum_{k \in N_{q_2}^+} I_{q_2, i, k} + x_{q_1, q_2, i}$$ \hspace{1cm} (52)

With the subgradients of the dual multipliers from Equation (50) - (52) and the step size from Equation (42), the dual multipliers are updated according to Equation (41). Then the updated dual multipliers are fed back to Equation (40) for the next iteration. The convergence rate depends on the choice of the step size. A large step size causes oscillation issues, while a small step size makes the convergence slow. Note that for computational speed
issue, we will not optimally solve the problem defined in Equation (39) but will terminate the dual ascent algorithm in a finite number of iterations and accept the best solution that is found. In Equation (41), $\alpha_r \partial g(\pi, \rho, \gamma)$, which is the increment on the dual multipliers, is determined by both the step size $\alpha_r$ and the subgradient in Equations (50) - (52). Besides the step size in Equation (42), another common choice of step sizes is $\alpha_r = \frac{1}{r}$. We perform preliminary experiments to compare these two choices of step size. In Equation (50) - (52), the value of $M$ influences the value of the subgradients. The impact of $M$ on the convergence rate is also studied in the experiments.

The test network presented in Liu and Dessouky (2017) is used in this paper to analyze the performance of the algorithm. In Figure 2, the convergence rate of the dual ascent method in different settings is studied. Both the value of $M$ and the step size formulation are studied in the experiment settings. $\text{step} = \text{norm}_{\text{inv}}$ denotes the normalized step size, as in Equation (42), which has the 2-norm of the dual multipliers in the denominator. The other step size setting is to choose the $\text{step} = 1/r$. We also compare the value of $M = 10,000$ with $M = 100,000$, since the value of $M$ is included in the calculation of the subgradient, as in Equations (50) - (52). As shown in Figure 2, the convergence of the normalized step size setting outperforms the convergence rate of $\text{step} = 1/r$, in both convergence speed and the objective value. A smaller value of $M$ improves the convergence rate at the early stage. The setting of $M = 10,000$ and $\text{step} = \text{norm}_{\text{inv}}$ is adopted throughout the rest of the experiments. Note that the convergence rate of the dual ascent method depends on the number of dual constraints. Based on these experiments, for this test network and schedule, we will use 30 as the maximum iteration number throughout the rest of the experiments.

![Figure 2: Convergence Rate of the Lagrangian Method](image-url)
4.1.2. Feasibility Recovery Algorithm

In Section 4.1.1, the dual ascent method is applied to solve the Lagrangian dual problem in Equation (38). The Lagrangian problem relaxes the precedence constraints (10), (11) and (12). Therefore, the solution from the dual ascent algorithm is not guaranteed to be feasible for the unrelaxed problem $Q^k_m$. Before passing the passenger train schedule to the subproblems, a recovery approach is needed to identify a feasible solution to $Q^k_m$. In this section, we present a heuristic to adjust the relaxed solution locally to fit the relaxed constraints and recover the feasibility of the solution.

Recall that the relaxed constraints guarantee the precedence relations and safety headway between trains. The violation of the constraints creates deadlock between trains. In the proposed model, deadlock happens if the precedence decisions indicate that train $q$ passes node $i$ before train $q'$ but the actual arrival time of $q'$ to node $i$ is no later than the departure time of $q$ from node $i$ plus a safety headway. Any violation of Constraints (10) and (11) generates deadlock. Also, route mismatch in Constraints (12) happens when the precedence decision sets $x_{q,q',i} = 1$ but the routes of the two trains do not overlap at node $i$. We define the deadlock and route mismatch as infeasibility in the solution.

A deadlock between two trains can be resolved in two ways, either by postponing one of the deadlocked trains and flipping the precedence decision, or by rerouting one of the trains to another node. We next discuss a heuristic that keeps the routing decisions unchanged, and identifies the minimal incremental cost when postponing one of the trains to resolve the deadlock. The route mismatch can be resolved by resetting the precedence decision variables properly.

Without loss of generality, we define that when a infeasibility occurs in the solution, the corresponding precedence decision is that train $q$ travels through node $i$ earlier than train $q'$, where $x_{q,q',i} = 1$. In the solution $\{x, I, t\}$, the infeasibility can be classified into two cases:

1. Route mismatch: The routes of train $q$ and train $q'$ do not intersect at node $i$.  
   \[ \sum_{j \in N_{i,q}^+} I_{q,i,j} + \sum_{k \in N_{i,q'}^+} I_{q',i,k} = 0 \]  
   (53)

2. Deadlock: The routes of train $q$ and train $q'$ intersect at node $i$.  
   \[ \sum_{j \in N_{i,q}^+} I_{q,i,j} = \sum_{k \in N_{i,q'}^+} I_{q',i,k} = 1 \]  
   (54)

Safety headway between train $q$ and train $q'$ is not guaranteed.  
\[ t_{q',i}^a < t_{q,i}^d + \mu \]  
(55)

In the second case, there is no trivial method that can recover feasibility of a solution, since the routes and schedules of all the trains are coupled in the solution. A systemic approach is needed to update the arrival and departure times of all the trains to guarantee that the re-timing does not introduce more infeasibilities to the solution. In Case 1, by flipping the precedence decision $x_{q,q',i}$, we can simply resolve the infeasibility made by Constraint (12). In Case 2, the cause of the infeasibility is that the safety headway is not guaranteed. Either keeping the precedence decision as $x_{q,q',i} = 1$ while postponing train $q'$, or flipping the precedence decision such
that \( x_{q,q',i} = 0 \) while postponing train \( q \) can increase the safety headway and resolve the deadlock. However, the two treatments postpone the different trains and lead the system to different states, thus the increments of the objective value function are not necessarily the same. The decision problem here is to identify the minimal incremental cost of the master problem objective when different treatments are applied to resolve a deadlock.

We propose a feasibility recovery algorithm to sequentially resolve the deadlocks and route mismatches from the solution. To identify the deadlocks and route mismatches that need to be resolved, a linear search through the constraints set of Equations (10), (11) and (12) can find the violated constraints. Note that after a recovery, the arrival and departure times of several trains could be updated and the status of the system changes. Thus, the infeasibilities are recovered sequentially and the algorithm should guarantee that no further infeasibilities are created when one infeasibility is resolved. Given a route of a passenger train \( q \) as the node set \( R_q = \{ i \mid \sum_{j \in N^{+}_{q,i}} I_{q,i,j} = 1, i \in N^t_q \} \), which remains unchanged in the feasibility recovery process, we start the recovery of infeasibilities from the upstream of the route. Node \( i \) is defined to be in the upstream of node \( j \) if both of them are in the route of train \( q \), and \( t^a_{q,i} < t^a_{q,j} \). After the infeasibility is recovered in the upstream node, the infeasibility in the downstream nodes might be recovered as well since the time decisions have been updated. For the efficiency of the algorithm, upstream nodes are preferred as the starting point for recovery rather than the downstream nodes.

Next we discuss an approach to recover the infeasibility. Recall that when different treatments are applied, the objective function incurs different incremental costs, which come from the additional delay of the postponed train. We denote the incremental delay as \( \epsilon_{q,i} \) after the infeasibility is resolved, by postponing train \( q \) at node \( i \). Note that after the infeasibility is resolved, all the arrival and departure times of train \( q \) along the downstream nodes are delayed by at least \( \epsilon_{q,i} \). Thus, \( t^a_{q,j} \) and \( t^d_{q,j} \) for train \( q \) on the downstream nodes \( j \) on its route will be updated. The most ideal case is that all of the downstream nodes of the postponed train \( q \) are available for entering at the updated arrival time. In this case, the constraints related to the postponed train is guaranteed to be feasible after the delay. However, most likely some of the downstream nodes are occupied at the delayed arrival time of train \( q \). Without resolving the conflict, the delayed arrival time could also cause a deadlock at downstream nodes. The feasibility recovery algorithm resolves all the deadlocks at downstream nodes as well. At all the downstream nodes of train \( q \) after node \( i \), we assign train \( q \) with low priority (train \( q \) yields to other conflicting trains) whenever deadlock happens. The rationality behind this is that more complexity will be introduced if the precedence decisions are flexible at the downstream nodes. The complexity comes from the fact that when a new deadlock happens at a downstream node \( j \), the other conflicting train \( q'' \) does not necessarily have the same travel direction with train \( q \). To resolve the new deadlock, node \( i \) could be the downstream node of node \( j \) for train \( q'' \), and the algorithm could revisit node \( i \) in a cycling decision process. We present Figure 3 as an example to illustrate how the complexity increases in the cycling deadlock case. In this double track network, suppose that when we resolve the deadlock at node \( i \), train \( q \) is postponed to minimize the incremental delay. At node \( j \), another train \( q'' \) is detected to conflict with train \( q \). Train \( q'' \) is in the opposite travel direction and nodes \( i \) and \( j \) are on the routes of both trains. To resolve the deadlock on node \( j \), any delay of train \( q'' \) might further create deadlock to train \( q \) back at node \( i \). This cycling decision process significantly increases the complexity of the algorithm. Therefore, the feasibility recovery algorithm assumes that the postponed train always has lower priority when conflicting with other trains on its downstream nodes.
Given a deadlock at node $i$ between train $q$ and train $q'$, to decide which train to postpone to resolve the deadlock, we will follow a "what if" process to first calculate what the incremental cost will be if we postpone train $q$ (or $q'$) and then adopt the better choice. If the infeasibilities come from Constraint (12), there is no incremental delay introduced at node $i$. The two routes are disjoint at node $i$, thus we don’t need to postpone any one of the trains. For the infeasibility from Constraints (10) or (11), the incremental delay $\epsilon_{q,i}$ comes from the postponement of train $q$ to be after train $q'$. We retrieve the arrival and departure time from the solution in Section 4.1.1 for the calculation.

$$\epsilon_{q,i} = t^{d}_{q',i} + \mu - t^{a}_{q,i}$$

(56)

Under this decision, the arrival time and departure time of train $q$ at node $i$ are updated. We denote the successor node of train $q$ after node $i$ on its route as node $i^+$.

$$t^{a}_{q,i} = t^{d}_{q',i} + \mu \quad \quad \quad \quad \quad (57)$$

$$t^{d}_{q,i} = t^{a}_{q,i} + B^{2}_{q,i,i^+} - B^{1}_{q,i} \quad \quad \quad \quad \quad (58)$$

At all the downstream nodes of the postponed train $q$ after node $i$ on its route $R_q$, train $q$ is assigned with a lower priority whenever it competes for a resource with another train. An algorithmic approach to generate the schedule is to start with constructing time windows for each node along the route $R_q$ to record the resource availability, and then identify the earliest exit time that train $q$ can leave a downstream node $j$ without influencing any other train’s schedule. As the trains are scheduled into the railway network, each track node may be occupied by some train during some period, and the occupied period is denoted as time windows. The time window is stored as a linked list data structure that automatically maintains a sorted and non-overlapping list of time windows when the node is occupied by the trains. When a node is assigned to the route of a train, the earliest exit time from the node depends on three factors: the earliest entering time $t^{e}_{q,j}$ of train $q$ at node $j$, the free flow travel time $B^{2}_{q,j,j^+}$, and the time windows of node $j$.

The objective of this step is to identify the first idle period among the time windows after the earliest entering time, and is sufficiently large for the train to travel through the node. The idle period is required to be sufficient for the free flow travel time $B^{2}_{q,j,j^+}$ plus twice the minimum safety headway $\mu$ since this guarantees the safety headway with the preceding train $q^-$ and the safety headway with the succeeding train $q^+$ (e.g., train $q^+ = q_3$). Train $q$ is scheduled to the end of the time window list if an adequate idle period is not identified.

We iteratively calculate the incremental delay $\epsilon_{q,j}$ and update the time decision on node $j$ based on node $j^-$. It is an iterative process along the downstream nodes of train $q$’s route $R_q$. First we introduce $t^{a}_{q,j}$ as the earliest time that train $q$ can enter node $j$. To resolve the infeasibility on node $j$, $t^{a}_{q,j}$ is calculated according to Equation
based on the departure time from the upstream node $j^-$. 

\begin{align*}
\epsilon_{q,j} &= t_{q,j}^d + \mu - \bar{p}_{q,j} \\
t_{q,j}^a &= t_{q,j}^d + \mu \\
t_{q,j}^d &= t_{q,j}^a + B_{q,j,j}^2 \\
\bar{p}_{q,j} &= t_{q,j}^d - B_{q,j,j}^2 + P_{q,j}^1
\end{align*}

After this iterative process finishes along the entire route of train $q$, the incremental cost of the master problem objective function is evaluated for the decision that postpones train $q$ at node $i$. The incremental cost is calculated as the weighted sum of the incremental delay $\epsilon_{q,j}$ between the stations. Denote the subset of nodes on route $R_q$ between station $i$ and station $j$ as $R_{q,i,j}$. Between every two stations $i, j \in S_q$, the incremental delays on the node set $R_{q,i,j}$ are summed up and then weighted by $p_{q,i,j}$. The incremental cost $c_q$ of postponing train $q$ is calculated as follows.

\begin{align*}
c_q &= \sum_{i,j \in S_q} (p_{q,i,j} \sum_{k \in R_{q,i,j}} \epsilon_{q,k}) \\
\end{align*}

The incremental cost is evaluated for two possible decisions on node $i$, one decision for postponing train $q$ and the other decision for postponing train $q'$. The case with less incremental cost is adopted and the decisions are applied to the solution, as shown in Equation (64). The precedence decisions $x_{q,q',i}$ and $x_{q',q,i}$ are set according to the updated arrival and departure times along the postponed train’s route. After these steps, we resolve the infeasibility at node $i$ and guarantee no further infeasibilities are introduced. The time complexity to resolve a infeasibility is $O(|N| \ast |Q_p|)$.

\begin{align*}
(x_{q,q',i}, x_{q',q,i}) &= \begin{cases} 
(1, 0) & \text{if } c_q \geq c_q' \\
(0, 1) & \text{if } c_q < c_q'
\end{cases}
\end{align*}

The relaxed solution from Section 4.1.1 may contain multiple infeasibilities. Our proposed overall infeasibility recovery algorithm consists of a sequential call to the single feasibility recovery approach. Each call is guaranteed to resolve at least one infeasibility. The detailed algorithm to resolve all the infeasibilities is as follows.

**Step 1:** Identify the conflict free trains in the solution, denoted as $\hat{Q}_p$. The trains that have infeasibilities to be resolved are denoted as $\bar{Q}_p = Q_p \setminus \hat{Q}_p$.

**Step 2:** For $q \in Q_p$, start from the infeasibilities on the upstream of its route, and sequentially resolve the infeasibilities on the route with the other trains in $\bar{Q}_p$.

**Step 3:** After all the infeasibilities by train $q$ are resolved, delete $q$ from $\bar{Q}_p$, and add it to $\hat{Q}_p$. If $\bar{Q}_p$ is not empty, go to Step 2. Else go to Step 4.

**Step 4:** Output the decision variables $(t, I, x)$ as a feasible solution to the master problem $Q_m$.

In this section, we presented the feasibility recovery algorithm to find a feasible solution to the master problem. The algorithm essentially applies a local adjustment of the solution towards the feasible domain. Optimality of the resulting solution is not guaranteed by the heuristic. However, the complexity of the algorithm is polynomial
with respect to the number of passenger trains and the number of nodes in the network. Due to the fact that the feasibility recovery algorithm is one step (Step 3.2) of our overall branch bound framework as presented in Section 4, it is important that this step is computationally fast since it is executed at every search tree node.

<table>
<thead>
<tr>
<th>Test case</th>
<th>Train index</th>
<th>Passenger demand</th>
<th>Test case</th>
<th>Train index</th>
<th>Passenger demand</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>(08:00, 300, 400)</td>
<td>2</td>
<td>1</td>
<td>(08:00, 350, 233)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>(09:00, 325, 425)</td>
<td></td>
<td>2</td>
<td>(08:20, 332, 789)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>(07:15, 500, 600)</td>
<td></td>
<td>3</td>
<td>(08:55, 100, 120)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>(08:15, 250, 350)</td>
<td></td>
<td>4</td>
<td>(10:15, 290, 390)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>(09:15, 375, 175)</td>
<td></td>
<td>5</td>
<td>(11:15, 115, 225)</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>(11:10, 130, 345)</td>
<td></td>
<td>1</td>
<td>(08:10, 324, 240)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>(12:10, 170, 225)</td>
<td></td>
<td>2</td>
<td>(09:00, 675, 695)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>(07:15, 650, 330)</td>
<td></td>
<td>3</td>
<td>(07:15, 720, 270)</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>(08:15, 780, 110)</td>
<td></td>
<td>4</td>
<td>(08:20, 550, 950)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>(09:15, 980, 1015)</td>
<td></td>
<td>5</td>
<td>(08:25, 345, 765)</td>
</tr>
</tbody>
</table>

Table 1: Test Cases for Passenger Train Schedules

The feasibility recovery algorithm recovers the feasibility of the solution by delaying some of the trains, and thus an extra cost is introduced to the objective function. The Lagrangian relaxed problem solved by the dual ascent algorithm gives a lower bound to the optimal solution of the original problem. Thus the gap of the master problem feasible solution is bounded by the Lagrangian relaxation. Next we experimentally show the bound on the optimality gap of the algorithm for a test network. Four passenger train schedules are generated to test the feasibility recovery algorithm in the same network as Section 4.1.1. Both the passenger demand \( p_{q,i,j} \) and the earliest departure time \( T^d_q \) are randomly generated. The generated demands and earliest departure times are listed in Table 1. The solution approach presented in this section is applied to the test network, and the Lagrangian solution, the solution after employing the feasibility recovery algorithm and the optimality gap are shown in Table 2. From the results, the upper bound of the optimality gap in the four test schedules is shown to be within 13 percent.

<table>
<thead>
<tr>
<th>Schedule</th>
<th>Lagrangian relaxation solution</th>
<th>Feasibility recovery solution</th>
<th>Optimality gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSG-1</td>
<td>31134</td>
<td>34127</td>
<td>9.6%</td>
</tr>
<tr>
<td>PSG-2</td>
<td>24910</td>
<td>27173</td>
<td>9.1%</td>
</tr>
<tr>
<td>PSG-3</td>
<td>41195</td>
<td>46239</td>
<td>12.2%</td>
</tr>
<tr>
<td>PSG-4</td>
<td>68917</td>
<td>77650</td>
<td>12.7%</td>
</tr>
</tbody>
</table>

Table 2: Optimality Gap of Feasibility Recovery Solution

4.2. Subproblem Solution Procedure

4.2.1. Labeling Algorithm for Rescheduling in a Scenario

In this section, we discuss the scheduling of the freight trains and the rescheduling of the passenger trains in the subproblem \( Q'_v \). The timetable extracted from the master problem solution is optimized to minimize the passengers’ travel times in Equation 1, assuming that there is no disturbance or other freight trains competing for the track resources. However, after introducing the freight trains in the subproblems, the track resource will be shared by all the trains. The weight \( \beta \) in Equation 18 defines the operation planner’s preference weight between the travel time of freight trains and the passenger train timetable deviation from the master problem.
The scheduling of freight trains and rescheduling of passenger trains depend on the freight trains departure times. The actual departure time of each freight train is sometimes uncertain due to the unexpected events such as locomotives maintenance, fueling, container loading time, etc. In the passenger train timetable planning problem, rather than assigning the freight trains with deterministic departure times, it is reasonable to consider the uncertainty of departure times and design the timetable to minimize the expected cost.

In this paper, the uncertainty in the departure time of each freight train is described as a discrete distribution, in which each departure time has a given probability. A scenario $v$ is a realization of the uncertainty where each freight train $q$ is assigned with a departure time $\omega^v_q$.

The subproblem itself is a large scale mixed integer problem considering the number of freight trains. A procedure that optimally solves the problem is not computationally feasible to solve $V$ subproblems where $V$ exponentially grows with the number of trains. In this section, we propose to construct an approximate optimal solution to the subproblems through a construction approach. The construction approach assigns the routes and schedules of the freight trains based on the departure times in scenario $v$ and the passenger train schedules, and also adjusts the passenger train schedules in order to maintain the feasibility while minimizing the objective value of the subproblem in Equation (18).

The construction approach consists of sequential scheduling of freight trains. During the process of route construction, it searches towards the direction that optimizes the subproblem by three kinds of decisions: route a freight train $q$, delay a freight train or delay a passenger train. In the sequential scheduling of trains, we assume that the previously scheduled freight trains before the processing of freight train $q$ are fixed in their routes and the rerouting is not allowed but the time and precedence decisions can be adjusted. The objective of individual freight train scheduling is to minimize the travel time of freight train $q$ while minimizing the impact to the other freight trains schedule and the planned passenger train timetable.

The route $R_q$ of train $q$ is a sequence of consecutive nodes, in which the first node is the origin station $O_q$ and the last node is the destination node $D_q$. Without considering the traffic congestion with other passenger trains and freight trains, the optimal route of an individual freight train is the shortest path over the network. However, the capacity limit of the trackage resource and the safety headway constraints introduce more complexity to the routing problem. In the objective function Equation (18), if the weight $\beta$ for passenger train timetable deviation is set large enough, the problem reduces to the shortest path problem with time windows and linear waiting cost, which is proved to be NP-complete in Desaulniers and Villeneuve (2000).

Consider the arrival of train $q$ at node $i$, an extra waiting time is generated if the arrival lies within an occupied time window of node $i$. The waiting time can be reduced either by postponing the other train $q'$ that currently occupies node $i$ to advance the feasible time window, i.e, allowing a shifting of the time windows, or by re-routing train $q$ to another track node $i'$ that has a shorter waiting time. In this study, shifting time windows is allowed and the schedule of a previously scheduled train can be adjusted.

Given the event that freight train $q$ arrives at node $i$, we define the arrival state as $z_{q,i} = (\Psi_{q,i}, \Delta_{q,i}, \Phi_{q,i})$, $i \in N_q^t$, where $\Psi_{q,i}$ is the sum of the freight train delays, and $\Delta_{q,i}$ is the sum of the passenger trains delays after the departure of train $q$ from node $i$. Time windows follow the same definition as in Section 4.1.2. The time windows state is denoted as $\Phi_{q,i}$, which is the combination of time windows on all network nodes at the departure of train $q$ from
The time windows state is the union of the lists, which is stored as a map data structure, from the nodes to the sortest lists of time windows. Initially, \( \Phi_{q,O_q} \) only contains the time windows of the passenger trains and the previously scheduled freight trains. With the expansion of \( z_{q,i} \), the time window state of each node is updated.

To optimize the route and schedule of freight train \( q \) in the shifting time windows state, both the delay of train \( q \) itself and the impact to the other trains are considered. We use the arrival state \( z_{q,i} \) to label the delays and the time windows states with the arrival of train \( q \) to node \( i \), and trace back the entire route when the labeled node set covers the whole node set \( N_t^q \) that are reachable by train \( q \).

Starting from the origin station \( O_q \), the labeling algorithm uses a breadth first search (BFS) to explore the candidate node set \( N_t^q \). Initially, \( \Phi_{q,O_q} \) is initialized with the time windows of passenger trains and previously scheduled freight trains. Then we iteratively consider the nodes in the unexplored node sets and label the nodes with an updated arrival state. When the labeling approach reaches an unexplored node \( i \), there are three states between freight train \( q \) and the previously scheduled trains. Under each state, there are several decisions that can be made.

**State 1:** During the travel time of train \( q \) at node \( i \), node \( i \) is congestion free.

- **Decision 1:** Schedule train \( q \) through node \( i \) immediately without any delay.

**State 2:** Node \( i \) is occupied by another passenger train \( q_p \).

- **Decision 2:** Train \( q \) enters node \( i \) after train \( q_p \) leaves, plus a minimum safety headway.

- **Decision 3:** Train \( q_p \) is postponed to wait until train \( q \) travels through node \( i \) first with a safety headway.

**State 3:** Node \( i \) is occupied by another previously scheduled freight train \( q_f \).

- **Decision 4:** Train \( q \) enters node \( i \) after \( q_f \) train leaves, plus a minimum safety headway.

- **Decision 5:** Train \( q_f \) is postponed to wait until train \( q \) travels through node \( i \) first with a safety headway.

For each decision, the *additional cost* of subproblem \( Q_v^w \) can be directly calculated according to the waiting time of each type of train. The additional cost for each decision case is summarized in Table 3. In the freight train travel time, both the waiting time and the traveling time are included.

Besides the incremental delay, another impact of the decision to schedule freight train \( q \) at node \( i \) is the change of time windows state \( \Phi_{q,i} \). Consider train \( q \), a new occupied time window is created at node \( i \), and thus the time window state is updated. Postponing a previously scheduled train changes the time window states of the corresponding nodes. The shifted time windows state may introduce infeasibilities to the schedule of the previously

<table>
<thead>
<tr>
<th>case</th>
<th>Freight trains travel time</th>
<th>Passenger trains delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision 1</td>
<td>( B_{q,i}^1 )</td>
<td>0</td>
</tr>
<tr>
<td>Decision 2</td>
<td>( t_{q,i}^d + \mu - t_{q,i}^a + B_{q,i}^1 )</td>
<td>0</td>
</tr>
<tr>
<td>Decision 3</td>
<td>( B_{q,i}^1 )</td>
<td>( t_{q,i}^d + \mu - t_{q,i}^a + B_{q,i}^1 )</td>
</tr>
<tr>
<td>Decision 4</td>
<td>( t_{q,i}^d + \mu - t_{q,i}^a + B_{q,i}^1 )</td>
<td>0</td>
</tr>
<tr>
<td>Decision 5</td>
<td>( t_{q,i}^d + \mu - t_{q,i}^a + B_{q,i}^1 )</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3: Incremental Delay in the Labeling Algorithm when Scheduling Train \( q \) on Node \( i \).
scheduled trains after the shift. Thus, the feasibility recovery algorithm in Section 4.1.2 is applied to check and resolve the infeasibilities after the update. Note that only the arrival and departure of the shifted time windows are necessary for feasibility recovery, thus the complexity of the feasibility recovery algorithm is further reduced. In the feasibility recovery process, the additional delays of both types of trains are non-negative. In the time windows state \( \Phi_{q,i} \), the incremental freight trains delay by the feasibility recovery is denoted as \( \psi(\Phi_{q,i}) \), and the incremental passenger trains delay is \( \delta(\Phi_{q,i}) \). The calculation of \( \psi(\Phi_{q,i}) \) and \( \delta(\Phi_{q,i}) \) is similar to the calculation in the feasibility recovery algorithm discussed in Section 4.1.2.

The routing of freight train \( q \) follows a labeling approach using the arrival state as the label. The updating of the label follows a dominate rule between the arrival states \( z_{q,i} = (\Psi_{q,i}, \Delta_{q,i}, \Phi_{q,i}) \) and \( \bar{z}_{q,i} = (\bar{\Psi}_{q,i}, \bar{\Delta}_{q,i}, \bar{\Phi}_{q,i}) \). Due to the flexible routing and different precedence decisions on the upstream nodes of node \( i \), there might be multiple candidate arrival states on one node for the train. The dominate arrival state is adopted as the label. We compare every two arrival states \( z_{q,i} \) and \( \bar{z}_{q,i} \), and define \( z_{q,i} \) dominates \( \bar{z}_{q,i} \) in the following dominate rule:

\[
\begin{align*}
\bar{z}_{q,i} \preceq z_{q,i} & \iff \\
\Psi_{q,i} + \psi(\Phi_{q,i}) + \beta[\Delta_{q,i} + \delta(\Phi_{q,i})] < \bar{\Psi}_{q,i} + \bar{\psi}(\bar{\Phi}_{q,i}) + \beta[\bar{\Delta}_{q,i} + \bar{\delta}(\bar{\Phi}_{q,i})]
\end{align*}
\]

Based on this dominate rule, the arrival state is updated if another dominate state is available at the arrival event. The label of each node in the node set \( N_q^t \) keeps updating with the expansion of the explored node set, until the whole node set \( N_q^t \) is covered and no more dominate states can be updated. We next present the update steps for arrival state \( z_{q,i} = (\Psi_{q,i}, \Delta_{q,i}, \Phi_{q,i}) \). Note that the freight trains are sequentially scheduled and the freight trains that are already scheduled are denoted as \( \bar{Q}_f \).

**Step 1:** Initialize

\[
\Phi_0 = \bigcup_{q \in Q_q} \{ [t_{q,i}^a, t_{q,i}^d + \mu, \forall i \in P_q] \}
\]

\[
\bar{z}_{q,O_q} = (\Psi_{q,O_q}, \Delta_{q,O_q}, \Phi_{q,O_q})
\]

\[
z_{q,i} = (\infty, \infty, \Phi_0), \forall i \in N_q^t \setminus O_q
\]

\[
t_{q,O_q}^d = \omega_q^O
\]

For the arrival state on the origin station, the initial state of \( \Psi_{q,O_q}, \Delta_{q,O_q} \) and \( \Phi_{q,O_q} \) inherits the state when the preceeding scheduled train arrives to its destination station.

**Step 2:** When train \( q \) enters successor node \( j \) from the preceding node \( i \) with state \( z_{q,i} = (\Psi_{q,i}, \Delta_{q,i}, \Phi_{q,i}) \), the state \( z_{q,j} \) based on the decisions can be calculated using Equation (70).

\[
z_{q,j} = \begin{cases} 
(\Psi_{q,i} + B_{q,j,i}^1, \Delta_{q,i}, \Phi_{q,j}) & \text{Decision 1} \\
(\Psi_{q,i} + t_{q,p,i}^d + \mu - t_{q,i}^a + B_{q,j,i}^1, \Delta_{q,i}, \Phi_{q,j}) & \text{Decision 2} \\
(\Psi_{q,i} + t_{q,p,i}^d, \Delta_{q,i} + t_{q,i}^d + \mu - t_{q,p,i}^a, \Phi_{q,j}) & \text{Decision 3} \\
(\Psi_{q,i} + t_{q,j,i}^d + \mu - t_{q,i}^a + B_{q,j,i}^1, \Delta_{q,i}, \Phi_{q,j}) & \text{Decision 4} \\
(\Psi_{q,i} + B_{q,j,i}^1 + t_{q,i}^d + \mu - t_{q,j,i}^a, \Delta_{q,i}, \Phi_{q,j}) & \text{Decision 5}
\end{cases}
\]
Step 3: The generated time windows state $\Phi_{q,j}$ differs according to which decision is applied. Under Decision 1, the update for the time window state is trivial, which is

$$\Phi_{q,j} = \Phi_{q,j} \cup [t^s_{q,j}, t^d_{q,j} + \mu]$$

(71)

In the case of Decision 2 to Decision 5, besides the newly added time window, the infeasibility recovery algorithm further adjusts the time windows state to shift the time windows on the related nodes if infeasibilities are detected. Thus, the update is determined iteratively.

Step 4: Repeat the labeling approach until the whole candidate node set $N^f_q$ is explored and the arrival state on destination node $D_q$ cannot be updated. Retrieve the entire route $R_q$ and the final state at the destination node $D_q$. The time windows state in $z_{q,D_q}$ is guaranteed to be feasible since each update is checked by the feasibility recovery algorithm. The final state $z_{q,D_q}$ is passed to the labeling approach of the next scheduled freight train as its initial state.

The labeling algorithm is applied to each of the freight trains sequentially. Thus, the overall approach generates an approximate optimal solution. Among the different scenarios, the departure times of the freight train are different. The scheduling sequence of all the freight trains in a scenario is ordered by the realization of the actual departure times. With each of the freight trains scheduled by the labeling algorithm, a scenario specific solution $(\bar{t}, \bar{I}, \bar{x})$ contains the schedules, routes and precedence relations of both freight trains and passenger trains. The passenger trains schedules are adjusted accordingly for each scenario by the labeling algorithm. The solution provides a feasible schedule for each scenario.

We apply the labeling algorithm to solve the freight train schedules and adjust the passenger train schedules in the test network. The value of $\beta$, which is the weight in the subproblem objective function, is included in the dominate rule to balance the priorities between passenger trains and freight trains. Experiments are applied to the test schedules introduced in Section 4.1.2. We randomly generate 20 scenarios of departure times of the freight trains at their origin stations. The discrete distributions of the departure times are presented in Table 4. In all the tests, we use the same 20 freight train scenarios to make the results consistently comparable.

<table>
<thead>
<tr>
<th>Freight train index</th>
<th>Departure time distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DISCRETE(07:41, 0.2; 07:43, 0.4; 07:44, 0.4)</td>
</tr>
<tr>
<td>2</td>
<td>DISCRETE(07:50, 0.5; 07:51, 0.5)</td>
</tr>
<tr>
<td>3</td>
<td>DISCRETE(07:49, 0.3; 07:51, 0.4; 07:52, 0.3)</td>
</tr>
<tr>
<td>4</td>
<td>DISCRETE(08:41, 0.7; 08:44, 0.3)</td>
</tr>
<tr>
<td>5</td>
<td>DISCRETE(08:51, 0.9; 08:53, 0.1)</td>
</tr>
<tr>
<td>6</td>
<td>DISCRETE(09:16, 0.2; 09:18, 0.6; 09:21, 0.2)</td>
</tr>
</tbody>
</table>

Table 4: Discrete Distributions of Freight Train Departure Times

We present the experimental results in Table 5, Table 6, and Table 7. Table 5 shows the average objective value of the subproblems in the 20 random scenarios. Table 6 and Table 7 show the breakdown of the subproblem objective value which comes from the freight train travel time component and the passenger train timetable deviation component. In Table 7, the deviation is compared between the master problem solution timetable and the adjusted timetable by the labeling algorithm. According to the experimental results, in each passenger schedule, with the increase of $\beta$, the average objective values of the subproblems increases. From Table 6, we observe that the main
increase in the subproblem objective value comes from the additional delay of the freight trains. With the increase of \( \beta \), the deviation of the passenger train timetable from the master solution timetable decreases. The value of \( \beta \) between 0.8 and 5.0 effectively guarantees the passenger timetable to be unaffected from the master problem timetable in the test schedules. With a small value of \( \beta \), the passenger trains yield to freight trains and the actual passenger trains schedules are delayed. A large \( \beta \) increases the priorities of the passenger trains and reduces the deviation of the actual timetable from the master problem timetable. The remaining experiments in this paper use \( \beta = 5 \).

<table>
<thead>
<tr>
<th>Passenger schedule</th>
<th>( \beta )</th>
<th>0.1</th>
<th>0.5</th>
<th>0.8</th>
<th>5</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSG-1</td>
<td>173.60</td>
<td>194.87</td>
<td>209.38</td>
<td>210.43</td>
<td>210.43</td>
<td></td>
</tr>
<tr>
<td>PSG-2</td>
<td>159.43</td>
<td>180.52</td>
<td>189.06</td>
<td>187.28</td>
<td>187.28</td>
<td></td>
</tr>
<tr>
<td>PSG-3</td>
<td>169.95</td>
<td>195.16</td>
<td>201.94</td>
<td>201.06</td>
<td>201.06</td>
<td></td>
</tr>
<tr>
<td>PSG-4</td>
<td>158.75</td>
<td>187.75</td>
<td>195.54</td>
<td>195.02</td>
<td>195.02</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Average Objective Values of the Subproblems Solved by the Labeling Algorithm

<table>
<thead>
<tr>
<th>Passenger schedule</th>
<th>( \beta )</th>
<th>0.1</th>
<th>0.5</th>
<th>0.8</th>
<th>5</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSG-1</td>
<td>154.30</td>
<td>193.35</td>
<td>209.38</td>
<td>210.43</td>
<td>210.43</td>
<td></td>
</tr>
<tr>
<td>PSG-2</td>
<td>153.92</td>
<td>175.31</td>
<td>189.06</td>
<td>187.28</td>
<td>187.28</td>
<td></td>
</tr>
<tr>
<td>PSG-3</td>
<td>153.20</td>
<td>192.27</td>
<td>201.21</td>
<td>201.06</td>
<td>201.06</td>
<td></td>
</tr>
<tr>
<td>PSG-4</td>
<td>151.91</td>
<td>184.71</td>
<td>194.74</td>
<td>195.02</td>
<td>195.02</td>
<td></td>
</tr>
</tbody>
</table>

Table 6: Average Total Freight Train Travel Time in the Subproblems

<table>
<thead>
<tr>
<th>Passenger schedule</th>
<th>( \beta )</th>
<th>0.1</th>
<th>0.5</th>
<th>0.8</th>
<th>5</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSG-1</td>
<td>191.94</td>
<td>3.06</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>PSG-2</td>
<td>55.06</td>
<td>10.41</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>PSG-3</td>
<td>167.51</td>
<td>5.78</td>
<td>1.52</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>PSG-4</td>
<td>68.44</td>
<td>6.07</td>
<td>1.64</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: Average Total Passenger Train Timetable Deviation from the Master Solution Timetable

### 4.2.2. Sample Average Approximation of All Scenarios

The objective value from the labeling algorithm solution contains two parts: the total travel time of the freight trains, and the total deviation from the timetable in the passenger train schedules. In the master problem objective function Equation (18), the expectation of the subproblem objective value \( E[h(t, I, \omega_v)] \) is a function of the master problem solution \((t, I, x)\). In the labeling approach, the passenger train schedule is modified in the adjusted schedules \((\tilde{t}, \tilde{I}, \tilde{x})\) for each random departure time vector \(\omega_v\) for scenario \(v\).

The random departure time vector \(\omega_v\) is the combination of the departure times of all the freight trains at their origin stations. The departure times of the freight trains are generated similarly to the discrete distributions that are discussed in Section 4.2.1. The total number of scenarios grows exponentially with respect to the number of freight trains. Considering the complexity of the algorithm, we approximate the expectation \( E[h(t, I, \omega_v)] \) by taking the sample average of a finite number of samples with size \(|V|\). \( \frac{1}{\sqrt{|V|}} \sum_{v=1}^{V} h(t, I, \omega_v) \) is calculated for each
scenario. Since the passenger trains’ schedules are adjusted, the master problems objective value changes as well. The Sample Average also includes the corresponding master problem objective values.

To provide a stable approximation, the sample size of the scenarios should be sufficiently large. As the number of scenario sample increases, the sample variance converges to the population variance but the computation time to calculate the approximation increases as well. Preliminary experiments are required to identify the minimum number of samples in which the sample variance converges. In the test network with 6 freight trains, we next study the relationship between the sample size, standard deviation of the subproblem objective values and the CPU time. The results are shown in Table 8. The CPU time increases linearly with the sample size, since each sample is solved by the same labeling algorithm. In this test case, a sample size between 30 and 50 gives a relative stable estimation since the standard deviation is close to convergence. Thus, we choose a sample size of 40 for the remaining experiments.

<table>
<thead>
<tr>
<th>Sample size</th>
<th>Standard deviation of objective values</th>
<th>CPU time(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>5.71</td>
<td>5.1</td>
</tr>
<tr>
<td>10</td>
<td>4.79</td>
<td>10.2</td>
</tr>
<tr>
<td>20</td>
<td>4.32</td>
<td>19.8</td>
</tr>
<tr>
<td>30</td>
<td>4.22</td>
<td>31.1</td>
</tr>
<tr>
<td>40</td>
<td>4.04</td>
<td>42.3</td>
</tr>
<tr>
<td>50</td>
<td>4.05</td>
<td>53.5</td>
</tr>
<tr>
<td>60</td>
<td>4.04</td>
<td>65.2</td>
</tr>
</tbody>
</table>

Table 8: Standard Deviation and CPU Time using Sample Average Approximation

With the selected sample size, the average master problem objective value is assigned as the objective value of the current search tree node $c(k)$. The value shows the best cost functions that our solution approach can reach given the master problem solution $(t, I, x)$, under the consideration of random freight train departure times. Without considering the freight trains, the master problem solution approach is biased to optimize the passenger trains schedule, and thus the master problem solution $(t, I, x)$ is insufficient to generate a well performed schedule for both passenger and freight trains. We leverage a branch and bound heuristic to explore the solution space of the master problem to improve the joint schedules of all the trains. Specifically, the search explores the choice of the passenger trains’ routes, and guides the passenger train to balance the load of the traffic through the underlying freight train traffic.

4.3. Branching Rules

In the labeling algorithm discussed in Section 4.2.1, the travel time of freight trains and the timetable deviation of passenger trains are balanced by $\beta$ which is introduced in the subproblem objective function. The balancing treatment only applies to the case when multiple trains request the same track resource at the same time. For example, when there is a conflict between a passenger train $q_p$ and a freight train $q_f$ on node $i$, the decision on postponing which train depends on the impact on the objective function. Some of the track nodes are preferred by both types of trains, i.e, the nodes with higher speed limit. In a general network with double tracks or multiple tracks, by greedily assigning the trains routes, the schedules tend to have local congestion on some commonly preferred track nodes.
In the solution framework we propose, the routes of the passenger trains are generated in the master problem solution approach. During the master problem solution approach, the freight trains demand is unknown and will be realized in the subproblems. Thus, there is no sufficient information to consider the freight trains when the passenger trains routing decision is made. Once the passenger trains routes are constructed, the subproblem labeling algorithm only adjusts the arrival/departure time decisions and precedence decisions, but the routes of the passenger trains remain unchanged. Thus the routing of passenger trains are biased towards the optimal passenger train schedule, rather than the optimal route for the total traffic.

To handle the drawback of this biased routing, we evaluate the traffic condition when solving the subproblems, and introduce the traffic condition as an extra constraint to the master problem to augment the solution with more routing constraints. In the branch and bound search framework, we implement a feed forward mechanic from the parent search tree node to the leaf nodes. Basically, all the leaf nodes are the copy of their parents node, except that each leaf node has a unique additional routing constraint. The additional routing constraint forbids the travel of a passenger train through a specific network node, which is treated as a congested node. In each search tree leaf node, the forbidden network node is set as unreachable for the train. We define the generative approach from the parent node to leaf nodes as branching. The branching is an iterative approach and the added routing constraint remains in the branch, until the branch reaches the infeasibility such that at least one train cannot reach the destination station from its origin station. The branching approach is shown in Figure 4. In the routing variable $I_{q,i,i^+}$, for passenger train $q$, $i^+$ is the succeeding node of $i$ on $q$’s route.

A breadth first search (BFS) is applied on the active node list $ANL$ with each search tree node evaluated by performing the master and subproblem solution approaches. The objective of the BFS search is to find the search tree node that has smallest objective value. The routes and timetable solved in the node is output as the final passenger trains’ decision. In this section, we discuss the criteria and process to construct a search tree starting from the root node to the leaf nodes.
4.3.1. Congestion Evaluation

The objective of the leaf nodes generation is to guide the passenger trains to detour the congested track node by adding more constraints. In this section, the heuristic to evaluate the traffic congestion encountered by passenger trains is discussed.

First, we define the *conflict* during the scheduling of a passenger train. A conflict happens in two cases when a passenger train requests a track node resource which is already occupied by another train, at the arrival time of the passenger train. The total number of conflicts is denoted as the *conflict count*. The conflict count reflects the frequency of requesting the track resource occupied by other trains. We use the conflict count of each passenger train at each node to evaluate the encountered traffic congestion.

Computationally, the conflict count is stored as a counter \( c_{qp,i} \) for the passenger train and node pair \((q_p, i), i \in N_q^t\). The counters are incremented during the execution of the labeling algorithm and the feasibility recovery algorithm, summed on all the evaluated subproblem scenarios. In the labeling algorithm, the new freight train conflicts with a passenger train in the Decision 2 and Decision 3 cases. The counters are incremented temporarily and then extracted using backtracking when the entire route is finalized. During labeling, the feasibility recovery algorithm adjusts the trains that create infeasibilities by postponing one of them. The postponed train updates the conflict count on its own route and also the other trains whose route shares the same node. In the feasibility recovery algorithm, the conflict counters are incremented when two trains share the same node, which is the case of Equation (54). The conflict count \( c_{qp,i} \) reflects the conflict decisions when the algorithm schedules train \( q_p \) to node \( i \). We develop a heuristic branching approach using \( c_{qp,i} \) and experimentally show that the conflict count can be used as a surrogate measure for congestion in the routing decision at the node level.

4.3.2. Heuristic Branching Strategy

We propose a branching strategy using the conflict count of each passenger train at each node to heuristically determine the passenger train routes. The objective of branching is to explore the domain space of the routing decisions and to minimize the cost function of the master problem under stochastic freight train departure times. Due to the resource capacity constraints and safety headway constraints, the track segments requested by multiple trains at the same time tend to generate the delay. By systematically scheduling trains and reducing the overlapping of the trains’ routing in the time space domain, local congestion can be avoided and the total delay of all the trains is reduced. Leveraging flexible routing and re-routing for trains for a complex network structure, the routes of passenger trains can be adjusted based on the observed local congestion which is evaluated by the conflict count.

The routing decision is initially made in the master problem solution approach. Considering the conflict count when solving the subproblems, a re-routing approach is implemented next to adjust the routing decisions and to address the traffic congestion issue.

Flexible routing for all passenger trains is a combinatorial optimization problem. To improve the route of the individual train and routes combination of all the trains, a neighborhood search can be implemented in the routing variable domain space around the current routing decision. A search that enumerates all the routing combinations is computationally expensive considering the size of the network and the number of trains. To heuristically improve the solution, we propose to branch the search tree along the direction that can potentially decrease the objective.
A route combination of trains that creates local congestion tends to increase the objective value. An intuitive example is that if every train is assigned to its shortest path, a track segment that is shorter or with a higher speed limit will be shared by multiple trains’ routes. In such a route combination, the delay significantly increases since some of the trains are waiting until the track resource is released. This congestion can be captured by the conflict count defined in Section 4.3.1. By rerouting the trains to detour the congested node, the waiting time at the congested track can be reduced.

In the detour process, which node is the most congested and which train is selected to detour the node needs to be decided. We propose a branching strategy to order the conflict count of all the trains at all the nodes, select the top three passenger trains and node pairs \((q_p, i)\) with the highest number of conflicts count, and generate a branched leaf node for each of the detour decisions. A constraint (see Equation (72)) is added back to the master problem, to generate an enhanced problem on the leaf node. The generated new search tree node is appended to the ANL. The solution algorithm on a search tree node that is discussed in Section 4.2 is applied to solve the generated problems.

\[
I_{q,i,i^+} = 0, \quad q \in Q_p, \hat{N}_q, j \in \hat{N}_{q,i}^+
\]  

(72)

The total number of problems to be solved grows exponentially with the branching process. Applying the solution algorithm to all the search tree nodes is computationally expensive and thus limits the search within a local solution space. We next present a bounding heuristic that trims the search tree branches to achieve better computational efficiency.

4.3.3. Bounding Rules

In the master problem solution approach, we aim to optimize the routing decisions, and adjust the time decisions and precedence decisions. Note that the constraints set (72) expands with the branch and bound process. A set of track nodes (empty in \(c(0)\)) is forbidden for some passenger trains to travel. As the process expands, the feasible domain of the passenger train routes shrinks until no feasible route exists for at least one passenger train. The master problem will eventually become infeasible if no available route exists for a passenger train between its origin and destination stations. If the individual train routing and scheduling problem in Equation (44) for any passenger train is infeasible, branching is terminated since all the leaf nodes become infeasible as well.

The branching heuristic reduces the sum of the delays by reducing the conflicts between the trains. However, rerouting can assign a suboptimal route to an individual passenger train. The cost of the total passenger travel time in the master problem does not necessarily decrease along a branch of the search tree. Not all the detour decisions are profitable for the overall problem, since the branching follows a heuristics rule. In the branching process, a bad detour decision will remain in the search tree once the detour constraint is added, and a significant portion of computation effort will be spent on exploring around the bad routing decision. We propose a straightforward bounding rule to discard the bad decisions before further branching, and the search in that direction will be terminated if the bounding criterion is met. When a node in \(ANL\) is solved, we record the objective value of the incumbent solution (best solution so far), denoted as \(obj(k^*)\), where \(k^*\) is the index of the search tree node that
has the incumbent solution. As discussed in the branching rule, the solution to the leaf node has one more detour constraint which makes the passenger train bypass the congested node. The leaf node solution does not necessarily outperform the solution of the parent node, since the detour decision may further generate delay for the detouring passenger train. We compare the objective value of the solved leaf node \( k \) with \( \text{obj}(k^*) \), as the following criterion:

\[
\text{obj}(k) > \text{obj}(k^*) \times (1 + \theta)
\]  

(73)

\( \theta \) is the bounding threshold, which defines the tolerance level for the change of objective value. Once the objective value exceeds the threshold, the heuristic bounding rule is applied to trim the current search tree node and to terminate the unpromising branches. The bounding rule possibly drops some high quality solutions but speeds up the search. The threshold addresses the unstable average approximation issue among the subproblems. As discussed in Section 4.2.2, the sample size is one of the factors that influences the variance of the samples. Given the sample size, the threshold needs to be determined.

The node that passes the bounding rule proceeds to the branching process. At the same time, the stopping criterion check is invoked to decide if the termination of the overall algorithm is reached. There are two stopping criteria that check the size of the \( \text{ANL} \) and the computation resource usage. If all the nodes in \( \text{ANL} \) do not meet the bounding condition and are removed from \( \text{ANL} \), there is no more search tree nodes that need to be explored. If the computation resource, including both computation time and memory, reaches the given limit, the branch and bound is also terminated. The incumbent solution to node \( k^* \) is output as the best solution.

The convergence rate of the branch and bound algorithm is analyzed through different experimental settings. The experiments are tested on the same network and schedule as in Section 4.2.2. First, we test random branching in which a randomly selected route constraint is generated to the leaf search tree nodes. In random branching, the congestion information from the subproblems is not considered and the branching direction is blind. Then we test the heuristic branching rule proposed in Section 4.3.2. The conflict count from the subproblems is treated as the estimation for the encountered traffic congestion. In this setting, the bounding rules are not applied. The experiment is denoted as \( \text{Branch \\& Bound, } \theta = \infty \). Then based on the heuristic branching, we further integrate the bounding rules with different bounding threshold values \( \theta \). The experiment \( \text{Branch \\& Bound, } \theta = 0.01 \) tests a lower bounding threshold, and the other experiment \( \text{Branch \\& Bound, } \theta = 0.05 \) tests a higher bounding threshold. The stopping criterion is set to be 200 branch and bound iterations. The convergence rates of the four experiments are shown in Figure 5, in which the objective values for each iteration is presented. Compared with random branching, the conflict-detour branching strategy effectively leads the search towards the direction that minimizes the objective. The value of \( \theta \) is shown to change the convergence rate of the conflict-detour branching algorithm. Compared with \( \theta = 0.01 \), the higher value of \( \theta = 0.05 \) relaxes the threshold and fails to trim some of the unpromising nodes, thus the objective value converges slower and is less stable. Thus, we will use \( \theta = 0.01 \) in the rest of the experiments.

5. Experimental Analysis

In this section, the branch and bound algorithm is experimentally studied. We compare the solution with other heuristics and analyze the robustness of the passenger train timetable. A part of the railway network from Corona to San Bernardino in California is considered in the test experiment. The railway network contains about
30 miles of trackage with complex track segments. The sketch of the railway network is presented in Figure 6. There are five passenger train stations in the network, and two freight train stations which are located at the beginning and ending point of the network. The translated abstract graph contains 120 nodes and 117 arcs. There are 10 passenger trains to be scheduled, and the passenger demand between stations are generated from a uniform distribution $\text{Unif}(40, 100)$. The earliest departure times for passenger trains from the origin stations are generated in a uniform-departure case and a compact-departure case. In the uniform-departure case, the earliest departure times of the passenger trains are uniformly distributed within the day. In the compact-departure case, the earliest departures times is scheduled within the rush hour of the day. There are 14 freight trains to be scheduled. The departure times of the freight trains follow some predefined discrete distributions. The uncertainty in departure times of a freight train is bounded within a limited range, as the test freight train departure time distributions shown in Table 9.

In Section 4.1.2, the bound on the optimality gap in passenger train schedule has been demonstrated through experiments. In this section, the experiment focuses on showing the robustness of the passenger train schedule, and the performance of the labeling algorithm in the jointly scheduling of passenger and freight trains when compared with other heuristics. First we defined the Passenger-\textit{OPT} schedule as the passenger train schedule that is optimal for the given passenger demand. This schedule is retrieved from the solution of the root search tree node $c(0)$, which has no conflict detouring constraints fed back from the subproblems. Without considering the
freight trains impact, \textit{Passenger-OPT} is biased towards passenger trains. After running the proposed branch and bound algorithm, the passenger train schedule in the incumbent solution to node $k^*$ is defined as \textit{Passenger-Robust}. The robust passenger train schedule accounts for the freight trains. Given the passenger train schedules, there are multiple strategies to the scheduling of freight trains. \textit{Greedy scheduling} is a greedy approach for scheduling and routing freight trains by routing the freight train to the succeeding node which has the earliest available resource. Prior to the availability, the freight train keeps waiting until the resource is free. Greedy scheduling keeps the passenger train schedules unchanged, and the routing decision is made only through the resource availability. \textit{Look-ahead scheduling} heuristic maintains an evaluation window to check several succeeding nodes. By enumerating the possible routes and schedules in a local range, the algorithm selects the best one and then moves to the succeeding node. \textit{Labeling algorithm scheduling} follows the labeling algorithm discussed in Section 4.2.1. The freight train scheduling approach jointly considers the delay of passenger trains. The labeling approach attempts to find the global optimal schedule and route for each freight train. By comparing the \textit{Passenger-OPT with Greedy} scheduling with \textit{Passenger-OPT with Look-ahead} scheduling, we study if the look-ahead heuristics can assign the freight trains with better routes. By comparing \textit{Passenger-OPT with Look-ahead} scheduling and \textit{Passenger-Robust with Look-ahead} scheduling, the robustness of the passenger trains timetable is evaluated. By comparing \textit{Passenger-Robust with Look-ahead} scheduling and \textit{Passenger-Robust with Labeling Algorithm} scheduling, we further study that

<table>
<thead>
<tr>
<th>Freight train index</th>
<th>Departure time distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DISCRETE(08:30, 0.2; 08:40, 0.5; 08:50, 0.3)</td>
</tr>
<tr>
<td>2</td>
<td>DISCRETE(10:27, 0.7; 10:35, 0.3)</td>
</tr>
<tr>
<td>3</td>
<td>DISCRETE(08:29, 0.3; 08:35, 0.4; 08:42, 0.3)</td>
</tr>
<tr>
<td>4</td>
<td>DISCRETE(10:47, 0.5; 10:57, 0.5)</td>
</tr>
<tr>
<td>5</td>
<td>DISCRETE(09:15, 0.4; 09:22, 0.5; 09:30, 0.1)</td>
</tr>
<tr>
<td>6</td>
<td>DISCRETE(10:01, 0.1; 10:11, 0.9)</td>
</tr>
<tr>
<td>7</td>
<td>DISCRETE(09:50, 0.1; 10:00, 0.5; 10:05, 0.4)</td>
</tr>
<tr>
<td>8</td>
<td>DISCRETE(10:20, 0.2; 10:30, 0.8)</td>
</tr>
<tr>
<td>9</td>
<td>DISCRETE(08:55, 0.6; 09:05, 0.3; 09:12, 0.1)</td>
</tr>
<tr>
<td>10</td>
<td>DISCRETE(10:45, 0.3; 10:55, 0.7)</td>
</tr>
<tr>
<td>11</td>
<td>DISCRETE(08:17, 0.2; 08:22, 0.7; 08:28, 0.1)</td>
</tr>
<tr>
<td>12</td>
<td>DISCRETE(09:08, 0.2; 09:12, 0.8)</td>
</tr>
<tr>
<td>13</td>
<td>DISCRETE(09:10, 0.6; 09:15, 0.1; 09:22, 0.3)</td>
</tr>
<tr>
<td>14</td>
<td>DISCRETE(10:33, 0.5; 10:39, 0.5)</td>
</tr>
</tbody>
</table>

Table 9: The Test Freight Trains Departure Time Distributions
given the robust passenger timetable, whether or not the proposed labeling algorithm outperforms the Look-ahead scheduling approach.

To test the performance of each algorithm, a set of randomly generated freight train departure times samples is computed and averaged. In each sample, the objective value contains the cost function of the master problem and of the subproblem. In Greedy scheduling and Look-ahead scheduling, the passenger train schedule remains unchanged when the freight trains are scheduled, thus the objective value of the master problem is constant and the deviation term in the subproblem is zero. Combining the passenger train schedule and the freight train scheduling strategies, we compare each objective value. In Table 10, Passenger Cost is the objective value of the master problem, which is the sum of the weighted passenger travel time. Timetable Deviation is the passenger train timetable deviation term in the objective function of the subproblem. Freight Cost is the total freight train travel time term in the objective function of the subproblem.

In Table 10, Passenger-Robust with Labeling Algorithm heuristic outperforms the other scheduling heuristic for the overall objective value. In Greedy freight train scheduling, the algorithm greedily routes the train to the node with the earliest available time. In a complex network, the routing tends to fall into local minimum solutions. Compared with the Greedy heuristic, the Look-ahead heuristics extends the local search strategy to find the best routes in a larger neighborhood of the solution space. By comparing Passenger-OPT with Greedy and Passenger-OPT with Look-ahead, the Look-ahead heuristic effectively reduces the freight trains cost. In these two experiments, the passenger trains timetable is optimized for passenger trains while not considering the encountered traffic congestion with freight trains and remains unchanged during the scheduling of the freight trains in the two heuristics. Comparing the freight cost of the two heuristics, Look-ahead reduces the total freight train travel time, and the reduction is more significant when the number of freight train increases. In the comparison between Passenger-OPT with Look-ahead and Passenger-Robust with Look-ahead, the robust passenger train timetable increases the passenger cost slightly. However, the overall objective of Passenger-Robust has better performance since the freight cost is improved in the robust passenger train timetable. In Passenger-Robust with Labeling approach, the robust passenger train timetable is further adjusted in the scheduling of freight trains. The passenger cost increases since some of the passenger trains are delayed compared to the initial robust timetable. However, the freight cost is further reduced since the priorities between all the trains are balanced such that the overall objective can be improved.

6. Conclusions

In this paper, the stochastic timetabling problem for passenger trains is studied considering the uncertainty in the freight train departure times. We presented a large scale two stage stochastic optimization model to leverage the flexible routing in the railway network and to optimize the operation cost given the uncertainties. The passenger demand between stations is considered in the first stage problem cost function. In the second stage problem, the passenger trains' schedules are adjusted based on the freight trains' departure times and the freight trains' schedules are optimized. We proposed a branch and bound solution framework to solve the problem efficiently. A BFS search is applied on the search tree. To solve the problem defined on each search tree node, we applied
<table>
<thead>
<tr>
<th># of Freight Trains</th>
<th>Overall Objective</th>
<th>Passenger Trains</th>
<th>Freight Trains</th>
<th>Overall Objective</th>
<th>Passenger Trains</th>
<th>Freight Trains</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Passenger Cost</td>
<td>Timetable Deviation</td>
<td>Freight Cost</td>
<td>Passenger Cost</td>
<td>Timetable Deviation</td>
<td>Freight Cost</td>
</tr>
<tr>
<td>6</td>
<td>20042.6</td>
<td>19670.9</td>
<td>0</td>
<td>371.7</td>
<td>19937.9</td>
<td>19580.1</td>
</tr>
<tr>
<td>8</td>
<td>20230.1</td>
<td>19670.9</td>
<td>0</td>
<td>559.2</td>
<td>20123.2</td>
<td>19580.1</td>
</tr>
<tr>
<td>10</td>
<td>20388.4</td>
<td>19670.9</td>
<td>0</td>
<td>717.5</td>
<td>20229.9</td>
<td>19580.1</td>
</tr>
<tr>
<td>12</td>
<td>20517.3</td>
<td>19670.9</td>
<td>0</td>
<td>846.4</td>
<td>20335.7</td>
<td>19580.1</td>
</tr>
<tr>
<td>14</td>
<td>20693.8</td>
<td>19670.9</td>
<td>0</td>
<td>1022.9</td>
<td>20471.3</td>
<td>19580.1</td>
</tr>
<tr>
<td></td>
<td>Passenger-Robust with Look-ahead</td>
<td>Passenger-Robust with Labeling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># of Freight Trains</td>
<td>Overall Objective</td>
<td>Passenger Trains</td>
<td>Freight Trains</td>
<td>Overall Objective</td>
<td>Passenger Trains</td>
<td>Freight Trains</td>
</tr>
<tr>
<td></td>
<td>Passenger Cost</td>
<td>Timetable Deviation</td>
<td>Freight Cost</td>
<td>Passenger Cost</td>
<td>Timetable Deviation</td>
<td>Freight Cost</td>
</tr>
<tr>
<td>6</td>
<td>19933.2</td>
<td>19591.1</td>
<td>0</td>
<td>342.1</td>
<td>19910.6</td>
<td>19599.3</td>
</tr>
<tr>
<td>8</td>
<td>20128.8</td>
<td>19591.1</td>
<td>0</td>
<td>521.7</td>
<td>20094.2</td>
<td>19602.7</td>
</tr>
<tr>
<td>10</td>
<td>20207.3</td>
<td>19591.1</td>
<td>0</td>
<td>616.2</td>
<td>20197.3</td>
<td>19609.2</td>
</tr>
<tr>
<td>12</td>
<td>20322.6</td>
<td>19591.1</td>
<td>0</td>
<td>731.5</td>
<td>20314.8</td>
<td>19615.1</td>
</tr>
<tr>
<td>14</td>
<td>20441.4</td>
<td>19591.1</td>
<td>0</td>
<td>860.3</td>
<td>20430.1</td>
<td>19619.6</td>
</tr>
</tbody>
</table>

Table 10: Comparison Between Passenger and Freight Trains Scheduling Strategies
heuristic algorithms to the master problem and subproblems. For the master problem, a dual ascent algorithm is employed to solve for a relaxed passenger trains schedule and then a feasibility recovery algorithm resolves the infeasibilities from the relaxed solution and generates a feasible passenger trains routing and schedule solution. A subproblem, which corresponds to a scenario of freight train departure times, is solved by a labeling algorithm by routing and scheduling the freight trains and adjusting the passenger trains schedules as well. Experimental analysis that compares the algorithm with other heuristic algorithms demonstrates the solution quality of our proposed approach.

7. Acknowledgement

We acknowledge Metrans and the Volvo Research and Education Foundation for its kind support of this research.

References


Yang, L., Qi, J., Li, S., & Gao, Y. (2016). Collaborative optimization for train scheduling and train stop planning on high-speed railways. *Omega, 64*, 57–76.