# Real-time Train Platforming and Routing at Busy Complex High-speed Railway Stations

Jia Ning [a,1], Qiyuan Peng [a,2], Gongyuan Lu [a,3]
[a] School of Transportation and Logistics, Southwest Jiaotong University, Chengdu, China
[1] E-mail: ningjia@my.swjtu.edu.cn
[2] E-mail: qiyuan-peng@home.swjtu.edu.cn
[3] E-mail: lugongyuan@home.swjtu.edu.cn, Phone: +0086-13880609100

**Abstract**
This paper focuses on the real-time train platforming and routing problem at a busy complex high-speed railway station in a disrupted situation caused by malfunctioning railway infrastructure or train primary delay. When the disruption occurs, dispatchers need to reassign trains to the conflict-free platform and route, even reschedule the arrival and departure times. To this end, we develop a mixed-integer linear programming formulation that determines platforming and routing decision simultaneously, while allowing trains to be rescheduled when the initial schedule imposes irreconcilable conflicts. The objective of our model is to minimize the overall deviation from the planned schedule and the original platforming plans. To improve the solving efficiency, an iterative algorithm is proposed to compute near-optimal solutions in a short computation time, which is based on the decomposition of the overall problem into two sub-problems: (i) platform and route assignment with fixed arrival and departure times, (ii) partial conflict trains rescheduling. The connecting information between two sub-problems concerns the index of conflict trains and the new train timetable. To solve sub-problem (i) efficiently, we develop a branch and bound algorithm which includes implicational rules enabling to speed up the computation and still can acquire optimal solutions. Since the model of sub-problem (ii) is the same as the model of original problem with a relative small scale, it can be solved by CPLEX solver efficiently. A real-world instance with operation data of Zhengzhou East high-speed railway station, is implemented to demonstrate the performance and effectiveness of the proposed algorithm.

**Keywords**
Train platforming and routing problem, Real-time conflict resolution, Linear programming, Branch and bound algorithm

## 1 Introduction

In this paper, we focus on an important operational problem in the railway industry, namely real-time train platforming and routing at a single large busy complex high-speed railway station (rtTPR). This problem is one variant of the more general problem of routing trains through railway stations considered by Zwaneveld et al. (2001).

Within a high-speed railway system, the platform, the route, as well as the arrival and departure time of each train are set in the dispatching system in advance, then trains run as scheduled. When the disruption occurs which can be caused by malfunctioning railway infrastructure or train primary delay, dispatchers need to reassign trains to the non-conflicting platforms and routes, even reschedule the arrival and departure times when the

initial schedule imposes irreconcilable conflicts.

Large stations, such as hub stations, are usually located at the intersection of multiple railway lines. Such stations typically have multiple entering/existing points, dozens of platforms and hundreds of inbound/outbound routes, with several hundred or over a thousand trains per day (Carey and Carville, 2003). Dispatchers should coordinate the station operations of trains from different entering/existing directions every day. When a disruption occurs, a high-quality platform reallocation plan can absorb train delay to some extent. This process requires effective solution within minutes, which is difficult for the dispatchers.

Most research on train dispatching is concerned with rescheduling trains on lines, usually single lines, taking into account the microscopic layout of each station on the railway line. But they are not concerned with large stations with multiple entering/existing points, or do not consider detailed route occupancy and release (for example, D'Ariano et al. 2008, Lamorgese and Mannino, 2015). However, in Europe and China, large busy complex high-speed railway stations are key components of high-speed railway networks, and are usually the locations of most train conflicts (Carey and Carville, 2003). Since the rtTPR is of great significance to reduce the impact of the disruptions on train operation and station working order, we focus on this problem in the present paper. And the optimization results can also provide suggestions for line dispatching.

The train platforming and routing at a single railway station (TPR) is an important planning problem in the railway industry and arises at each of the strategic, tactical and operational planning levels (Sels et al., 2014). While the strategic and tactical level variants, which address future station capacity and the generation of feasible timetable, respectively, have been well studied, the operational variant has received relatively limited attention in the literature. Lusby et al. (2011) survey a large number of papers in the field of routing trains through railway junctions at the strategic, tactical and operational levels. Cacchiani et al. (2014) present an overview of literatures on real-time timetable rescheduling in case of disturbances or disruptions, considering a microscopic or a macroscopic view on the railway system. We refer the interested reader to Lusby et al. (2011) and Cacchiani et al. (2014). Here we review contributions in the area of real-time train platforming and routing at a single station. Without providing an exhaustive review, we attempt to provide an overview of the models and methods that have been adopted in the literature.

Zwaneveld et al. (1996) address the problem of routing trains through railway stations at the strategic level. In this paper the authors propose a node-packing formulation which allows time deviations on the arrival and departure times of train paths. Furthermore, they present a branch-and-cut approach combined with reduction techniques to solve the problem. However, in a follow-up paper, Zwaneveld et al. (2001) state that this approach was unable to solve the routing problem for two of the larger stations in Netherlands. Zwaneveld et al. (2001) is an extension of Zwaneveld et al. (1996). In this paper the problem is formulated as a weighted node-packing model. More sophisticated preprocessing and reduction techniques are included in the branch-and-cut solution approach, and the authors conclude that this approach is sufficient for solving the routing problem arising at any Dutch railway station, but the efficiency of this approach cannot be applied to the operational level.

Carey and Carville (2003) address the TPR at the tactical level. They develop scheduling heuristics analogous to those successfully adopted by dispatchers using "manual'' methods. The algorithm considers one train at a time and finds and resolves all conflicts for that train before considering the next train. When considering a train, the

algorithm considers assigning it to each platform in turn, to find the best platform. With successive refinements, the algorithm eventually takes only a few seconds to run.

Constraint programming has also been used to model the TPR. Rodriguez and Kermad (1998) and Rodriguez (2007) are an attempt to model the operational variant of the problem. This approach attempts to find the minimum total delay necessary in keeping the trains on their assigned paths. Instances with between 6 and 24 trains are considered. However, the problem considered in this paper permits trains to wait on track sections if the subsequent track section is unavailable, which is not common in China and is not allowed in our paper.

D'Ariano et al. (2007, 2008) propose an alternative graph formulation for the operational variant of the TPR problem. A pair of alternative arcs is used to enforce a train sequencing order at any block section where two trains are in conflict. An alternative graph is built using one path per train. The model may include hundreds of machined (block sections) and jobs (trains) for real-life instances, and is therefore very hard to be solved in real-time. To overcome this issue, D'Ariano et al. (2007) propose a branch-and-bound algorithm which includes dynamic and static implication rules enabling to speed up the computation. This algorithm is extended by D'Ariano et al. (2008) to include a local re-routing strategy. The iterative procedure first computes an optimal train sequencing for given train routes and then improves this solution by locally rerouting some trains.

Caimi et al. (2012) propose a closed-loop discrete-time control framework for the TPR at the operational level. This framework resolve conflicts by re-timing and re-routing of trains as well as partial speed profile coordination. In this approach the time horizon is discretized, and a binary variable is associated with every operation and every period in the time horizon. Computational experiments indicate clearly the great potential of this approach.

Lusby et al. (2011) address the strategic-level variant of the TPR and present a set-packing model. A resource based constraint system is used, in which resources correspond to track sections. Then the authors prove that this formulation is tighter than the conventional node-packing model and develop a branch-and-price algorithm that utilizes the dual representation of any basic feasible solution. Lusby et al. (2013) extend this method and apply it to the operational level. In this paper, the authors develop a branch-and-price approach that exploits the flexibility of the model to be dynamically updated. Numerical results indicate the efficiency of this approach by confirming that, with a given time limit of 270 seconds, practical problems can be solved within 3.5 percent of optimality.

Caprara et al. (2011) deal with a general version of the TPR problem. Each train to be assigned a platform is assumed to have a number of possible patterns consisting of an inbound path, outbound path, and platform, as well as arrival and departure times at the platform. The authors present an integer linear programming formulation that is based on a node-packing problem. This model has a continuous relaxation that leads to strong bounds on the optimal value. A branch-and-cut-and-price solution approach based on the linear programming relaxation is proposed in this paper.

Boccia et al. (2013) present a new mixed integer programming model to tackle the real-time railway traffic management problem. A set of routes for each train is considered and tracks are subdivided into sections. The model uses binary variables indicating whether a route in the set is assigned to a train, and continuous variables representing the time at which a train reaches a block section. Two heuristic algorithms are proposed.

The same problem is considered by Meng and Zhou (2014). They propose a cumulative flow variables-based model based on a time-space network modelling

framework. A Lagrangian relaxation solution framework is developed. Then the original problem is decomposed into a sequence of single train optimization sub-problems. For each sub-problem, a dynamic programming algorithm is proposed to find the time dependent least cost path on a time-space network.

The similar problem is considered by Pellegrini et al. (2015, 2019). Pellegrini et al. (2015) present a mixed-integer linear programming formulation, which models the infrastructure in terms of track-circuits and the route-lock sectional release interlocking system. The model calls for assigning a route to each train, as well as a possible delay for the train on each track circuit. Since the difficulty in solving the MILP model is mostly due to the multiplicity of both the alternative routes and the potential conflicts. Pellegrini et al. (2015, 2019) propose valid inequalities to boost the solution algorithm. In addition, Pellegrini et al. (2019) reformulate this model based on a reduced number of scheduling binary variables.

Sama et al. (2016) deal with the real-time train rerouting and rescheduling in the railway network. This paper studies the problem of selecting the best subset of routing alternatives for each train among all possible alternatives, which is formulated as an integer linear programming formulation and solved via an algorithm inspired by the ant colonies' behaviour. Then, the real-time train rerouting and rescheduling problem takes as input the best subset of routing alternatives and is solved as a mixed-integer linear program.

In this paper, we focus on the rtTPR. The *Degree of Conflict* is defined to describe the conflict between any inbound/outbound routes. A bi-objective mixed integer linear programming model is formulated to determine platform, inbound and outbound route, and arrival and departure times simultaneously, which is also a universal model for the route-lock sectional/integral release interlocking system. Similar to the normal practice of railway dispatchers, this model is decomposed into two sub-models and an iterative algorithm which combines a branch-and-bound algorithm and CPLEX solver is developed. Finally, a real-world instance of Zhengzhou East high-speed railway station in China is tested.

## 2 Problem Description

A railway station consists of platforms and of a large number of track sections. Trains enter a railway station from *entering points* and leave it through *leaving points*. An *inbound route* is a sequence of track sections linking an entering point to a platform; while an *outbound route* is a sequence of track sections linking a platform to a leaving point. Notably, there may be more than one inbound/outbound route linking the same entering/leaving point and the same platform. A *path* is composed of an inbound and an outbound route linking the same platform and the linked platform, and it is a sequence of track sections connecting an entering point to a leaving point. There are generally multiple different paths between a given pair of entering and leaving points.

As soon as a train arrives at its entering point of the station, it claims an inbound route to a platform. At the same time, the linked platform is also claimed. Similarly, before a train leaves its platform, it claims an outbound route to its leaving point. Moreover, as a train traverses its inbound/outbound route, it sequentially releases each of the track sections comprising the route. Since any track section can only be claimed by at most one train at any time, a *conflict* will occur if two chosen routes simultaneously attempt to claim the same track sections. Thus, the exact calculation of the time at which the common sections are released by the previous route is the key to rule out the conflict

between any two routes. To this end, Degree of Conflict (DOC) is defined to describe the conflict relations between routes.

**Definition1. (Degree of Conflict)** Given a route pair $r - r'$, if route $r$ is claimed ahead of route $r'$, and route $r$ and $r'$ have common track sections, DOC between the route pair $r - r'$, denoted as $\gamma_{r,r'}$, indicates the time elapsed from the time when route $r$ starts to be claimed until the common sections are all released by route $r$ (i.e., the time at which route $r'$ can be claimed).

As shown in Figure 1, the sequenced sections list of route $r$ are recorded as {$s_9$, $s_8$, $s_7$, $s_4$, $s_3$, $s_2$, $s_5$, $s_6$}, and the sequenced sections list of route $r'$ are recorded as {$s_1$, $s_2$, $s_3$, $s_4$, $s_{10}$}. The common sections of route $r$ and $r'$ are recorded as {$s_2$, $s_3$, $s_4$}. If route $r$ is claimed ahead of route $r'$, then $\gamma_{r,r'}$ is equal to the duration of traversing sections {$s_9$, $s_8$, $s_7$, $s_4$, $s_3$, $s_2$} sequentially. Whereas, if route $r'$ is claimed ahead of route $r$, then $\gamma_{r',r}$ is equal to the duration of traversing sections {$s_1$, $s_2$, $s_3$, $s_4$} sequentially. Therefore, the DOC between routes is asymmetry (i.e., $\gamma_{r,r'} \neq \gamma_{r',r}$).

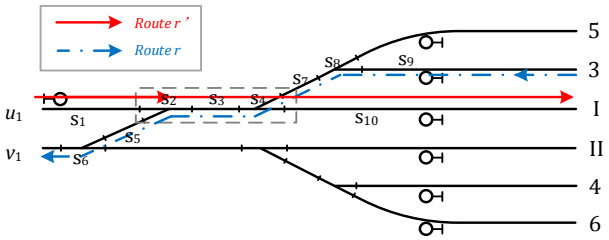


Figure 1: Illustration of Degree of Conflict.

Assume that all trains travel at the same constant speed regardless of route choice. DOC between each route pair can be calculated according to the sequenced sections list of each route, the common sections list, the length of each section and the speed of trains.

In addition to conflicts between routes, a conflict also arises whenever two or more trains require the same platform at the same time. To rule out platform conflicts, safety time interval is imposed between two adjacent trains occupying the same platform.

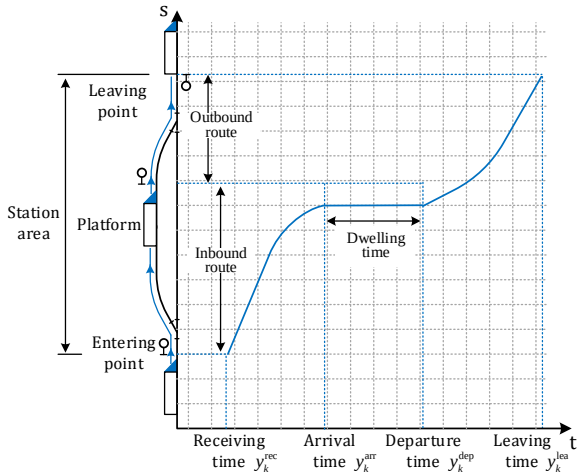


Figure 2: Time definition in train operation process.

For recording traveling process in railway station of each train, following variables are defined:

1. **Receiving time $y_k^{\mathbf{rec}}$:** The receiving time of train $k$ is the time at which the train arrives at its entering point of the station and its inbound route and platform have just been claimed.
2. **Arrival time $y_k^{\mathbf{arr}}$:** The arrival time of train $k$ is the time at which the train stops at a platform, after traveling along an inbound route.
3. **Departure time $y_k^{\mathbf{dep}}$:** The departure time of train $k$ is the time at which the train begins to leave the platform along an outbound route.
4. **Leaving time $y_k^{\mathbf{lea}}$:** The leaving time of train $k$ is the time at which the tail of the train releases the last section of its outbound route and leaves the station from its leaving point.

Notably, if a train does not stop at a railway station, which is called through train for the station, when it arrives at its entering point, its outbound route should be claimed with its inbound route and its platform. Thus, as for the through train, its receiving time indicates the time when the train arrives at its entering point of the station and its inbound and outbound route and platform have all been claimed. The arrival time of a through train indicates the time when the train has released the last section of its inbound route. And the departure time of a through train is equal to its arrival time. In addition, the definition of the leaving time of a through train is same as that of other trains.

Note that above 4 types of time definition correspond to the operation process of a train in a railway station, as illustrated in Figure 2, so relation can be modelled between these time variables. The arrival time is equal to the receiving time plus the time required by a train to release its inbound route completely (including the time for creating the inbound route, driver response time, the time for the train to approach the protection signal, the travel time of the train via the route, and the time for the train to clear and release the last section of the inbound route). Similarly, the leaving time is equal to the departure time plus the time required by a train to release its outbound route completely.

## 3 Mathematical Formulation

### 3.1 Assumptions

Firstly, the following assumptions are made throughout this paper.

**Assumption 1.** All trains travel at the same constant speed regardless of route choice. The DOC between each route pair and the traversal time of each route are pre-given.

**Assumption 2.** The ideal arrival and departure times of the trains derive from the planned scheduled at the tactical level. The original platform allocation plan and route assignment plan are per-given. The initial train delay information and track maintenance information are also pre-given.

### 3.2 Definitions

The sets, parameters and decision variables used in this paper are described in the Tables 1 and 2, respectively.

Table 1: Definition of sets and parameters.

| Symbol | Definition |
|---|---|
| $K$ | Set of trains, index by $k$ and $h$, i.e., $k, h \in K$. |
| $T$ | Set of platforms. |
| $R$ | Set of routes, index by $r$ and $r'$, i.e., $r, r' \in R$. |
| $R_k$ | Set of train paths that train $k$ potentially travel through, index by $i$ and $j$, i.e., $i, j \in R_k$. Each path $i$ is composed of three parts: inbound route $r_{k,i}^{\text{in}}$, platform $r_{k,i}$ and outbound route $r_{k,i}^{\text{out}}$, and $r_{k,i} \in T$, $r_{k,i}^{\text{in}}, r_{k,i}^{\text{out}} \in R$. |
| $\gamma_{r,r'}$ | The Degree of Conflict between route $r$ and route $r'$. |
| $w_k$ | 0-1 train parameter, equal to 1 if train $k$ is a through train, 0 otherwise. |
| $\tau_k^{\text{arr}}$ | Ideal arrival time of train $k$. |
| $\tau_k^{\text{dep}}$ | Ideal departure time of train $k$. |
| $\Delta_k$ | Minimum dwelling time of train $k$. |
| $c_{k,i}$ | Weight of train $k$'s path $i$. |
| $tt_{k,i}^{\text{in}}$ | Traversal time of inbound route $r_{k,i}^{\text{in}}$. |
| $tt_{k,i}^{\text{out}}$ | Traversal time of inbound route $r_{k,i}^{\text{out}}$. |
| $\theta_{k,i}$ | 0-1 availability parameter, equal to 1 if platform $r_{k,i}$ is available during the whole considered time horizon, 0 otherwise. |
| $t_{k,i}^{\text{start}}$ | Start time of the maintenance on platform $r_{k,i}$. |
| $t_{k,i}^{\text{end}}$ | End time of the maintenance on platform $r_{k,i}$. |
| $\text{H}_{\text{start}}$ | Start time of considered time horizon. |
| $\text{H}_{\text{end}}$ | End time of considered time horizon. |
| $\varepsilon$ | The shortest time interval permitted between two adjacent trains occupying the same platform. |
| M | A sufficiently large positive number. |

Table 2: Definition of variables.

| Symbol | Definition |
|---|---|
| $x_{k,i}$ | 0-1 path selection variable, equal to 1 if train $k$ chooses path $i$, 0 otherwise. |
| $y_k^{\text{arr}}$ | Actual arrival time of train $k$. |
| $y_k^{\text{dep}}$ | Actual departure time of train $k$. |
| $y_k^{\text{rec}}$ | Actual receiving time of train $k$. |
| $y_k^{\text{lea}}$ | Actual leaving time of train $k$. |
| $B_{k,i}$ | 0-1 occupancy-maintenance sequence variable, equal to 1 if train $k$ claiming platform $r_{k,i}$ precedes the maintenance on platform $r_{k,i}$, 0 otherwise. |
| $B_{k,h}^{\text{p}}$ | 0-1 platform occupancy sequence variable, equal to 0 if train $k$ claiming its platform precedes train $h$, 1 otherwise. |
| $B_{k,h}^{\text{in-in}}$ | 0-1 inbound-inbound route occupancy sequence variable, equal to 0 if train $k$ claiming its inbound route precedes train $h$ claiming its inbound route, 1 otherwise. |
| $B_{k,h}^{\text{in-out}}$ | 0-1 inbound-outbound route occupancy sequence variable, equal to 0 if train $k$ claiming its inbound route precedes train $h$ claiming its outbound route, 1 otherwise. |
| $B_{k,h}^{\text{out-in}}$ | 0-1 outbound-inbound route occupancy sequence variable, equal to 0 if train $k$ claiming its outbound route precedes train $h$ claiming its inbound route, 1 otherwise. |
| $B_{k,h}^{\text{out-out}}$ | 0-1 outbound-outbound route occupancy sequence variable, equal to 0 if train $k$ claiming its outbound route precedes train $h$ claiming its outbound route, 1 otherwise. |

### 3.3 Mathematical Model

The mathematical model for real-time Train Platforming and Routing problem is given in

the following.

**Objective function**

On one hand, the reallocation of platform and the reassignment of route will influence regular station working order. Hence, the first objective maximizes the preferences of the adjusted platform allocation plan in order to minimize the impact on regular station working order. This objective function can be formulated as follows:

$$\max Z_1 = \sum_{k\in K}\sum_{i\in R_k} c_{k,i}x_{k,i} \tag{1}$$

Here $c_{k,i}$ reflects the impact of choosing path $i$ for train $k$. A higher value of $c_{k,i}$ indicated a smaller impact on regular station working order.

On the other hand, in order to prevent train delay from propagating through the network, the second objective minimizes the overall deviation from the ideal arrival and departure times for each train. This objective function can be functioned as follows:

$$\min Z_2 = \sum_{k\in K}(y_k^{\mathrm{arr}} - \tau_k^{\mathrm{arr}}) + \sum_{k\in K}(y_k^{\mathrm{dep}} - \tau_k^{\mathrm{dep}}) \tag{2}$$

**Constraints**

(1) Time relation constraints.

The time variables of the model include actual receiving time, actual arrival time, actual departure time and actual leaving time. Constraints (3) specify that the actual arrival time of train $k$ is equal to its actual receiving time plus the time required to release its inbound route completely. Similarly, constraints (4) specify that the actual leaving time of train $k$ is equal to its actual departure time plus the time required to release its outbound route completely.

$$y_k^{\mathrm{arr}} = y_k^{\mathrm{rec}} + \sum_{i\in R_k} tt_{k,i}^{\mathrm{in}} x_{k,i} \qquad \forall k \in K \tag{3}$$

$$y_k^{\mathrm{lea}} = y_k^{\mathrm{dep}} + \sum_{i\in R_k} tt_{k,i}^{\mathrm{out}} x_{k,i} \qquad \forall k \in K \tag{4}$$

(2) Platform conflict free constraints

Each platform can be occupied by at most one train at any time. Whether it is a through train or a non-through train, its platform is claimed at its receiving time and is released at its departure time. Constraints (5) and (6) impose the safety time interval between two adjacent trains $k$ and $h$ on the same platform. One of the constraints (5) and (6) will be activated if both the variables $x_{k,i}$ and $x_{h,j}$ are equal to 1 and platform $r_{k,i}$ and $r_{h,j}$ refer to the same platform. The activation of a constraint (5) means train $k$ precedes train $h$ and thus $B_{k,h}^{\mathrm{p}}$ is equal to 0. In this case, the time when train $h$ claims its allocated platform minus the time at which train $k$ releases its allocated platform must be larger than or equal to the shortest time interval permitted between two adjacent trains occupying the same platform. Constraints (6) have a similar function of constraints (5) except that train $h$ precedes train $k$.

$$\mathrm{M}\left(2 + B_{k,h}^{\mathrm{p}} - x_{k,i} - x_{h,j}\right) + y_h^{\mathrm{rec}} - y_k^{\mathrm{dep}} \geq \varepsilon \qquad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, r_{k,i} = r_{h,j} \tag{5}$$

$$\mathrm{M}\left(3 - B_{k,h}^{\mathrm{p}} - x_{k,i} - x_{h,j}\right) + y_k^{\mathrm{rec}} - y_h^{\mathrm{dep}} \geq \varepsilon \qquad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, r_{k,i} = r_{h,j} \tag{6}$$

(3) Route conflict free constraints

Whether it is a through train or a non-through train, its inbound route is claimed at its receiving time and is released at its arrival time. However, as for the through train, its outbound route is claimed at its receiving time and is released at its leaving time; while as for the non-through train, its outbound route is claimed at its departure time and is

released at its leaving time.

The DOC of a given route pair indicates the duration time from the time when the previous route is claimed to the time when the latter route can be claimed. Constraints (7)-(14) impose minimum required headway times (i.e., the Degree of Conflict) between any two trains on two conflicting routes. It is worth noting that if a train is a through train, its outbound route conflicts with a route chosen by the other train, and the train claiming its outbound route precedes its conflict route, then the minimum required headway time is equal to the DOC between the two conflict routes plus the time required to release the through train's inbound route. Constraints (7) and (8) deal with conflicts between inbound routes. Constraints (9)-(12) deal with conflicts between inbound routes and outbound routes. Constraints (13) and (14) deal with conflicts between outbound routes.

$$\mathrm{M}\left(2 + B_{k,h}^{\mathrm{in-in}} - x_{k,i} - x_{h,j}\right) + y_h^{\mathrm{rec}} - y_k^{\mathrm{rec}} \geq \gamma_{r_{k,i}^{\mathrm{in}}, r_{h,j}^{\mathrm{in}}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{k,i}^{\mathrm{in}}, r_{h,j}^{\mathrm{in}}} \neq 0 \quad (7)$$

$$\mathrm{M}\left(3 - B_{k,h}^{\mathrm{in-in}} - x_{k,i} - x_{h,j}\right) + y_k^{\mathrm{rec}} - y_h^{\mathrm{rec}} \geq \gamma_{r_{h,j}^{\mathrm{in}}, r_{k,i}^{\mathrm{in}}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{h,j}^{\mathrm{in}}, r_{k,i}^{\mathrm{in}}} \neq 0 \quad (8)$$

$$\mathrm{M}\left(2 + B_{k,h}^{\mathrm{in-out}} - x_{k,i} - x_{h,j}\right) + w_h y_h^{\mathrm{rec}} + (1 - w_h) y_h^{\mathrm{dep}} - y_k^{\mathrm{rec}} \geq \gamma_{r_{k,i}^{\mathrm{in}}, r_{h,j}^{\mathrm{out}}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{k,i}^{\mathrm{in}}, r_{h,j}^{\mathrm{out}}} \neq 0 \quad (9)$$

$$\mathrm{M}\left(3 - B_{k,h}^{\mathrm{in-out}} - x_{k,i} - x_{h,j}\right) + y_k^{\mathrm{rec}} - w_h y_h^{\mathrm{rec}} - (1 - w_h) y_h^{\mathrm{dep}} \geq \gamma_{r_{h,j}^{\mathrm{out}}, r_{k,i}^{\mathrm{in}}} + w_h t t_{h,j}^{\mathrm{in}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{h,j}^{\mathrm{out}}, r_{k,i}^{\mathrm{in}}} \neq 0 \quad (10)$$

$$\mathrm{M}\left(2 + B_{k,h}^{\mathrm{out-in}} - x_{k,i} - x_{h,j}\right) + y_h^{\mathrm{rec}} - w_k y_k^{\mathrm{rec}} - (1 - w_k) y_k^{\mathrm{dep}} \geq \gamma_{r_{k,i}^{\mathrm{out}}, r_{h,j}^{\mathrm{in}}} + w_k t t_{k,i}^{\mathrm{in}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{k,i}^{\mathrm{out}}, r_{h,j}^{\mathrm{in}}} \neq 0 \quad (11)$$

$$\mathrm{M}\left(3 - B_{k,h}^{\mathrm{out-in}} - x_{k,i} - x_{h,j}\right) + w_k y_k^{\mathrm{rec}} + (1 - w_k) y_k^{\mathrm{dep}} - y_h^{\mathrm{rec}} \geq \gamma_{r_{h,j}^{\mathrm{in}}, r_{k,i}^{\mathrm{out}}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{h,j}^{\mathrm{in}}, r_{k,i}^{\mathrm{out}}} \neq 0 \quad (12)$$

$$\mathrm{M}\left(2 + B_{k,h}^{\mathrm{out-out}} - x_{k,i} - x_{h,j}\right) + w_h y_h^{\mathrm{rec}} + (1 - w_h) y_h^{\mathrm{dep}} - w_k y_k^{\mathrm{rec}} - (1 - w_k) y_k^{\mathrm{dep}} \geq \gamma_{r_{k,i}^{\mathrm{out}}, r_{h,j}^{\mathrm{out}}} + w_k t t_{k,i}^{\mathrm{in}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{k,i}^{\mathrm{out}}, r_{h,j}^{\mathrm{out}}} \neq 0 \quad (13)$$

$$\mathrm{M}\left(3 - B_{k,h}^{\mathrm{out-out}} - x_{k,i} - x_{h,j}\right) + w_k y_k^{\mathrm{rec}} + (1 - w_k) y_k^{\mathrm{dep}} - w_h y_h^{\mathrm{rec}} - (1 - w_h) y_h^{\mathrm{dep}} \geq \gamma_{r_{h,j}^{\mathrm{out}}, r_{k,i}^{\mathrm{out}}} + w_h t t_{h,j}^{\mathrm{in}} \quad \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h : h > k, \gamma_{r_{h,j}^{\mathrm{out}}, r_{k,i}^{\mathrm{out}}} \neq 0 \quad (14)$$

(4) Path availability constraints

Some train paths may be unavailable due to track maintenance. Depending on the location of the track maintenance, it may incur the unavailability of some inbound routes, some outbound routes or some platforms during a predetermined time period, and then leads to the unavailability of some train paths. In this paper, we take platform maintenance as an example.

If a platform needs to be maintained, constraints (15) and (16) ensure that the time period of any train occupying the platform cannot overlap with the time period of the maintenance on the platform. When train $k$ selects path $i$, if $B_{k,i}$ is equal to 0, constraint (15) is used to ensure that train $k$ can only claim the platform $r_{k,i}$ after the maintenance on this platform has been executed; otherwise, constraint (16) enforces that train $k$ must exit

from the platform $r_{k,i}$ before the start of maintenance on this platform.

$$\mathrm{M}\left(1+B_{k,i}-x_{k,i}\right)+y_k^{\mathrm{rec}}-t_{k,i}^{\mathrm{end}}\geq 0 \quad \forall k\in K,\forall i\in R_k\colon \theta_{k,i}=0 \tag{15}$$

$$\mathrm{M}\left(2-B_{k,i}-x_{k,i}\right)+t_{k,i}^{\mathrm{start}}-y_k^{\mathrm{dep}}\geq 0 \quad \forall k\in K,\forall i\in R_k\colon \theta_{k,i}=0 \tag{16}$$

(5) Earliest arrival/departure time constraints

To guarantee passengers' boarding activity, each train is not permitted to depart earlier than its ideal departure time. Moreover, since in this paper, we reschedule trains without considering the train movement in line sections and other adjacent stations, to guarantee that the adjusted train schedule is also feasible on the whole network, each train is not permitted to arrive earlier than its ideal arrival time.

$$y_k^{\mathrm{arr}}\geq \tau_k^{\mathrm{arr}} \quad \forall k\in K \tag{17}$$

$$y_k^{\mathrm{dep}}\geq \tau_k^{\mathrm{dep}} \quad \forall k\in K \tag{18}$$

(6) Minimum dwelling time constraints

The time required by passengers to board and alight dictates the minimum amount of dwelling time required. For each train $k$, constraint (19) ensures that its actual dwelling time is larger than or equal to its minimum dwelling time $\Delta_k$. Obviously, the minimum dwelling time is set to 0 if train $k$ is a through train.

$$y_k^{\mathrm{dep}}-y_k^{\mathrm{arr}}\geq \Delta_k \quad \forall k\in K \tag{19}$$

(7) Path selection constraints

Each train $k$ can select exactly one path.

$$\textstyle\sum_{i\in R_k} x_{k,i}=1 \quad \forall k\in K \tag{20}$$

(8) Domain of variables

The domain of variables in the model is defined by expressions (21)-(23) and is next summarized. The actual receiving time, the actual arrival time, the actual departure time and the actual leaving time of each train are defined as integer variables. The rest of the variables are defined as binary variables.

$$y_k^{\mathrm{rec}},y_k^{\mathrm{arr}},y_k^{\mathrm{dep}},y_k^{\mathrm{lea}}\in N \quad \forall k\in K \tag{21}$$

$$x_{k,i},B_{k,i}\in\{0,1\} \quad \forall k\in K,\forall i\in R_k \tag{22}$$

$$B_{k,h}^{\mathrm{p}},B_{k,h}^{\mathrm{in-in}},B_{k,h}^{\mathrm{in-out}},B_{k,h}^{\mathrm{out-in}},B_{k,h}^{\mathrm{out-out}}\in\{0,1\} \quad \forall k\in K,\forall h\in K \tag{23}$$

The proposed model is a mixed-integer linear programming formulation that can be solved by commercial solvers. However, solver efficiency of the model is still a matter in large scale problem solving due to 3 aspects of issues: (1) two types of conflict (platform and route) need to be resolved separately which expand the scale of the model; (2) sequence of trains are set as decision variable; and (3) arrival and departure times may need to be rescheduled once conflict occurs.

Thus, the following section aims to develop a heuristic algorithm that can efficiently obtain a high-quality solution in a much short time for the model. Next, we firstly decompose the overall problem into two sub-problems and then detailed techniques of the algorithm are introduced.

## 4 Solution Approaches

### 4.1 Decomposition of MILP Model

When platform/route conflict occurs, dispatchers generally first consider reassigning trains to the conflict-free paths. If the conflict still cannot be resolved, then dispatchers will modify the arrival and departure times of the relevant trains. Based on the normal practice of dispatchers, the real-time Train Platforming and Routing problem can be

decomposed into two sub-problems: (i) train path selection sub-problem with fixed arrival and departure times (TPSWFT sub-problem), (ii) partial conflict trains rescheduling sub-problem (PCTR sub-problem).

Carey and Carville (2003) also develop a heuristic algorithm which is analogous to the "manual'' methods adopted by dispatchers. The algorithm considers one train at a time and finds and resolves all conflicts for that train before considering the next train. Although this algorithm takes only a few seconds to run, it may facilitate the propagation of train delay on the railway network.

Lamorgese and Mannino (2015) decompose the real-time train dispatching problem into two sub-problems (i.e., line dispatching sub-problem and station dispatching sub-problem). The line dispatching sub-problem attempts to reschedule trains in order to minimize the deviations from the original timetable; the station dispatching sub-problem is the train platforming feasibility problem based on a given timetable. Compared with the above paper, the PCTR sub-problem in this paper is used to reschedule trains in real time and the TPSWFT sub-problem is used to assign non-conflicting platform and routes to each train. The decomposition approach is similar to the decomposition approach proposed by Lamorgese and Mannino (2015). However, the TPSWFT sub-problem in this paper is the train platforming optimization problem, while the station dispatching sub-problem is the train platforming feasibility problem. And a high-quality platform reallocation plan can absorb train delay to some extent. In addition, when solving the station dispatching sub-problem, Lamorgese and Mannino (2015) only consider the trains from or to two specific entering/existing points, while the station considered in this paper usually have multiple entering/existing points, we can collaboratively optimize the allocated platform (and inbound and outbound route) and arrival and departure times of all trains from or to different entering/existing points.

Dollevoet et al. (2014) consider the problem of delay management. They propose an iterative heuristic which first solves the delay management model with a fixed platform track assignment and then improves this platform track assignment in each step. However, for the rtTPR, the impact of rescheduling trains on train operations is more severe than the impact of reassigning trains to new platforms and routes, thus the strategy of reassigning trains to new platforms and routes should be given priority.

**TPSWFT sub-problem**

This sub-problem attempts to reallocate conflict-free paths (composed of inbound routes, platforms, and outbound routes) for as many trains as possible without modifying the arrival and departure time of each train. The paths are selected to minimize the impact on regular station working order. The arrival and departure time of each train are taken from the initial train schedule and train delay information or updated by PCTR sub-problem. The TPSWFT formulation is as follows:

TPSWFT: $\max Z = \sum_{k\in K} \sum_{i\in R_k} c_{k,i} x_{k,i}$

Subject to:

Constraints (3)-(16), (21)-(23)

$$y_k^{\mathrm{arr}} = \tau_k^{\mathrm{arr}} \quad \forall k \in K \tag{24}$$

$$y_k^{\mathrm{dep}} = \tau_k^{\mathrm{dep}} \quad \forall k \in K \tag{25}$$

$$\sum_{i\in R_k} x_{k,i} \leq 1 \quad \forall k \in K \tag{26}$$

**PCTR sub-problem**

This sub-problem aims to further resolute platform and route conflicts through

rescheduling the arrival and departure times of trains. The set of trains involved in solving this sub-problem, denoted by $K'$, includes all the trains that cannot be allocated a conflict-free path in the TPSWFT sub-problem. The model of the PCTR sub-problem is similar to the model of the original problem, except that the train sets considered by these two problems are different. The PCTR formulation is as follows:

PCTR: $\min Z = \sum_{k\in K'}(y_k^{\mathrm{arr}} - \tau_k^{\mathrm{arr}}) + \sum_{k\in K'}(y_k^{\mathrm{dep}} - \tau_k^{\mathrm{dep}})$

Subject to:

Constraints (3)-(23)

### 4.2 Solution Approach of TPSWFT Sub-problem

When the arrival and departure times of each train are fixed, the conflict relationship between any two paths of different trains can be determined. Let $\delta_{k,i,h,j}$ denote whether the path $i$ of train $k$ conflicts with the path $j$ of train $h$, which is equal to 0 when the two paths conflict with each other (because of platform conflict or route conflict), and 1 otherwise. In addition, the binary parameter $\delta_{k,i,h,j}$ has the following characteristics:

(1) Symmetry, i.e., $\delta_{k,i,h,j} = \delta_{h,j,k,i}, \forall k \in K, \forall h \in K, \forall i \in R_k, \forall j \in R_h$.
(2) If the path $i$ of train $k$ is unavailable, then $\delta_{k,i,h,j} = 0$ and $\delta_{h,j,k,i} = 0$, $\forall h \in K, \forall j \in R_h$.
(3) Since each train can be assigned to at most one path, $\delta_{k,i,k,j} = 0(\forall k \in K, \forall i,j \in R_k: i \neq j)$ and $\delta_{k,i,k,j} = 1(\forall k \in K, \forall i,j \in R_k: i = j)$.

An undirected conflict graph $G = (V, A)$ is built based on the conflict relationships between train paths, where each vertex $v_{k,i} \in V$ corresponds to a possible path $i$ for train $k$ and is assigned a weight $c_{k,i}$, and each arc $a_{k,i,h,j} \epsilon A$ connecting the two vertexes (vertex $v_{k,i}$ and vertex $v_{h,j}$) indicates that the corresponding train paths (path $i$ of train $k$ and path $j$ of train $h$) are compatible with each other (i.e., when $\delta_{k,i,h,j}$ is equal to 1, the arc $a_{k,i,h,j}$ exists). It is worth noting that there is no connection between any vertexes corresponding to paths for the same train.

The TPSWFT sub-problem attempts to reallocate conflict-free paths for as many trains as possible. Based on the undirected conflict graph, this sub-problem can be formulated as the Maximum Vertex Weight Clique Problem (MVWCP). The MVWCP can be described as follows:

> Given an undirected graph $G = (V, A)$, a clique is a set $C \subseteq V$ such that there is exactly one arc connecting any two vertexes of $C$. And for a clique $C$ of $G$, define its weight as $W(C) = \sum_{v_{k,i}\in C} c_{k,i}$. The MVWCP is to determine a clique $C^*$ of maximum weight, i.e., $\forall C \in \Omega, W(C^*) \geq W(C)$ where $\Omega$ is the set of all possible cliques of the graph.

Furthermore, for each vertex $v_{k,i}$ of $G$, the *vertex weight degree* $wd_{k,i}$ is defined to reflect the maximum weight that the clique may reach if vertex $v_{k,i}$ is selected. The formula for calculating $wd_{k,i}$ is as follows:

$$wd_{k,i} = \sum_{h\in K} \max\{\delta_{k,i,h,j} c_{h,j} | j \in R_h\} \quad (27)$$

Many algorithms and methods have been proposed to solve MVWCP, see Wu and Hao (2015). In this paper, we develop a branch and bound algorithm to solve it which includes implicational rules enabling to speed up the computation and still can acquire optimal solutions.

The branch and bound algorithm is developed in the form of a tree structure. Each level of the tree, denoted by $l$, represents assigning path for train $l$ ($l \neq 0$). Each node on any particular level, denoted by $n_{l,p}$, represents assigning path $p$ for train $l$ ($p \leq |R_l|$) or indicates that no path can be assigned for the train ($p = |R_l| + 1$). Leaf nodes define feasible train path selection plans or partial maximum clique.

For each node $n_{l,p}$ of the branch-and-bound tree, the following variables are defined:

(1) the current clique $C_{l,p}$ on node $n_{l,p}$ is used to record all chosen vertexes currently.
(2) the current weight $cw_{l,p}$ on node $n_{l,p}$ is defined to reflect the accumulated weight of the current clique.
(3) the upper bound $ub_{l,p}$ on node $n_{l,p}$ is defined to reflect the maximum weight that the partial maximum clique may reach if the branch is continued based on node $n_{l,p}$ until one leaf node is obtained. The formula for calculating $ub_{l,p}$ is as follows:

$$ub_{l,p} = \sum_{h \in K} \max\{c_{h,j} \times \min\{\delta_{k,i,h,j} | v_{k,i} \in C_{l,p}\} | j \in R_h\} \quad (28)$$

(4) the conflict relationship matrix $E_{l,p}$ on node $n_{l,p}$ is defined to reflect whether any path of any train conflicts with any chosen path of the current clique. Each element $e_{k,i}^{l,p}$ of matrix $E_{l,p}$ can be computed by formula (29).

$$e_{k,i}^{l,p} = \min\{\delta_{k,i,h,j} | v_{h,j} \in C_{l,p}\} \quad (29)$$

**Branch and Bound algorithm procedure**

**Step 0.** Initialization. Set $l = 0$ and $p = 1$, and generate the root node $n_{0,1}$. Set $C_{0,1} = \emptyset$ and $cw_{0,1} = 0$. Each element $e_{k,i}^{o,1}$ of matrix $E_{0,1}$ is set to 1. The upper bound $ub_{0,1}$ on the root node $n_{0,1}$ is also the upper bound of the overall TPSWFT sub-problem, denoted by $UB$, which can be computed by formula (30). Turn to Step 1.

$$UB = ub_{0,1} = \min\{\max\{wd_{k,i} | i \in R_k\} | k \in K\} \quad (30)$$

**Step 1.** Node selection. If all nodes of the branch-and-bound tree are leaf nodes, the branch and bound algorithm terminates; otherwise, pick the node of the last level with the maximum current weight and the maximum upper bound, turn to Step2.

**Step 2.** Branching and Bounding. Based on the selected node in Step 1, assign a path for the next train, and the vertex in the conflict graph corresponding to the specified path is added into the current clique of the selected node accordingly. Hence, generate a series of nodes, and the number of newly generated nodes is equal to the number of possible paths of the next train plus one. For each newly generated node $n_{l,p}$, calculate $C_{l,p}$, $cw_{l,p}$, $ub_{l,p}$ and $E_{l,p}$, and turn to Step 3. When the last newly generated node has been checked, turn to Step 1.

**Step 3.** Pruning. For each newly generated node $n_{l,p}$, (1) if the newly added vertex is not connected to each vertex in the current clique of the selected node, node $n_{l,p}$ will be removed; (2) if $l = |K|$, i.e., node $n_{l,p}$ is a leaf node, and if the current weight $cw_{l,p}$ is greater than the current optimal solution, then update the current optimal solution. And if the current optimal solution is equal to $UB$, the branch and bound algorithm terminates; otherwise, for each node $n_{m,q}$ of the branch-and-bound tree, if its upper bound $ub_{m,q}$ is less than or equal to the current optimal solution, then node $n_{m,q}$ is removed; (3) if $l < |K|$ and the upper bound $ub_{l,p}$ is less than or equal to the current optimal solution, then node $n_{l,p}$ is removed; (4) if $l < |K|$

and the upper bound $ub_{l,p}$ is greater than the current optimal solution, turn to Step 4 to check whether node $n_{l,p}$ meets *Equivalence Rule*.

**Step 4.** Equivalence Rule. For the newly generated node $n_{l,p}$ and any node $n_{l,q}$ which is on the same level and still exists on the branch-and-bound tree after Step 3, if the matrix $E_{l,p}$ and the matrix $E_{l,q}$ are equivalent, which shows that for any leaf node derived by continuous branching based on node $n_{l,p}$, there must be a leaf node derived by continuous branching based on node $n_{l,q}$, and these two leaf nodes have the same weight, then node $n_{l,p}$ is removed. The conditions for the equivalence of matrix $E_{l,p}$ and matrix $E_{l,q}$.are described as follows:

$\forall h \in K, h > l$ and $\forall j \in R_h$,

(1) if $r_{h,j} = r_{l,p}$, then

$$\begin{cases} e_{h,j}^{l.p} = 0, e_{h,j}^{l.q} = 0, & \text{if } \forall jj \in R_h, r_{h,jj} \neq r_{l,q} \\ e_{h,j}^{l.p} = e_{h,jj}^{l.q}, & \text{if } \exists jj \in R_h, r_{h,jj} = r_{l,q} \end{cases} \tag{31}$$

(2) if $r_{h,j} = r_{l,q}$, then

$$\begin{cases} e_{h,j}^{l.p} = 0, e_{h,j}^{l.q} = 0, & \text{if } \forall jj \in R_h, r_{h,jj} \neq r_{l,p} \\ e_{h,j}^{l.p} = e_{h,jj}^{l.q}, & \text{if } \exists jj \in R_h, r_{h,jj} = r_{l,p} \end{cases} \tag{32}$$

(3) if $r_{h,j} \neq r_{l,p}$ and $r_{h,j} \neq r_{l,q}$, then

$$e_{h,j}^{l.p} = e_{h,j}^{l.q} \tag{33}$$

In conclusion, a key strategy in the reduction of the computational effort of branch and bound algorithm procedures for the TPSWFT sub-problem is that based on the concept of the *vertex weight degree*, high quality upper bound can be obtained which serves both as an efficient pruning strategy, as well as an efficient stopping criterion. In addition, the *Equivalence Rule* is also used to tremendously reduce the size of the branch-and-bound tree as a more efficient pruning strategy. Based on these above implicational rules, the branch and bound algorithm enable to acquire optimal solutions within short time limits.

### 4.3 Solution Approach of PCTR Sub-problem

The trains which cannot be allocated a conflict-free path in the TPSWFT sub-problem, should be rescheduled in the PCTR sub-problem. At the same time, paths will be reassigned for the conflict trains in order to minimize the overall deviation from the ideal planed schedule. Since the model of the PCTR sub-problem is similar to the model of the original problem with a relative small scale, it can be solved by CPLEX solver efficiently. The algorithm for solving the PCTR sub-problem, which is called synchronous adjustment algorithm, is described as follows:

**Synchronous adjustment algorithm procedure**

**Step 0.** Generate conflict trains set $J$. The unassigned trains set $K'$ is the set of trains which cannot be allocated a conflict-free path in the TPSWFT sub-problem. For each train $k$ of set $K'$, calculate its conflict trains set $J_k$. If the station occupancy time of train $k$ (i.e., from its receiving time to its leaving time) overlaps with the station occupancy time of train $h$ ($\forall h \in K$), which implies that the changes of the arrival and departure times of train $k$ may cause the infeasible path of train $h$ or rescheduling the arrival and departure times of train $h$ may rule out the conflict between train $k$ and other trains, then train $h$ is added into the set $J_k$. And $J =$

$\{J_k | \forall k \in K'\}$. Turn to Step 1.

**Step 1.** Set operations. $\forall k \in K'$, $\forall h \in K'$, and $k \neq h$, if $J_k \cap J_h \neq \emptyset$, then make $J_k = J_k \cup J_h$, remove $J_h$ from $J$. Turn to Step 2.

**Step 2.** Synchronous adjustment. For each conflict trains set $J_k$, call PCTR model to further resolve the conflicts.

### 4.4 Iterative Algorithm

Since not all trains will input to the PCTR sub-problem, the feasibility is not guaranteed for that rescheduled trains may lead to new conflicts. Therefore, these two sub-problems need to be solved iteratively until all conflict are resolved.

In addition, to ensure that the iterative algorithm stops within the time limit, we use an additional criterion to terminate the algorithm, i.e., if the current iteration index is greater than a pre-given threshold, the iterative algorithm terminates. Specifically, $\chi$ denotes the iteration index, and $N$ denoted the maximum number of iterations.

When the algorithm terminates, if there are still some conflicts between the trains, for each unassigned train, assign it to its original allocated platform and inbound and outbound routes, and delay it until its arrival and departure times are greater than each of the assigned trains on its original allocated platform and make sure it is compatible with all other trains. Thus a feasible solution is obtained.

The iterative algorithm framework is as follows:

**Iterative algorithm procedure**

**Input:** The ideal planed train schedule, original platform allocation plan, original route assignment plan, detailed station yard topology, track maintenance information, initial train delay information, and so on.

**Step 0.** Initialization. Generate all possible paths for each train and the Degree of Conflict between any two routes. Set iteration index $\chi = 0$, and turn to Step 1.

**Step 1.** TPSWFT sub-problem. Calculate the binary parameter $\delta_{k,i,h,j}$, construct the undirected conflict graph, and call the branch and bound algorithm to reallocate conflict-free paths for as many trains as possible. If the number of unassigned trains is equal to 0, the iterative algorithm terminates. Set $\chi = \chi + 1$, if $\chi \leq N$, then turn to Step 2; otherwise, generate a feasible solution, and the iterative algorithm terminates.

**Step 2.** PCTR sub-problem. Collect all unassigned trains in set $K'$, and call the synchronous adjustment algorithm to further resolute conflicts through rescheduling the arrival and departure times of trains. Turn to Step 3.

**Step 3.** Update the arrival and departure times of each train and the conflict relationship between any two paths of different trains. Turn to Step 1.

**Output:** The adjusted arrival and departure times of each train, the adjusted path selection plan (including platform reallocation plan and the route reassignment plan).

## 5 Case Study

We performed the numerical experiment using operational data from the Zhengzhou East high-speed railway station to test how well the proposed algorithm may be applied in the real-world instance. The following experiment is performed on a computer Intel® Core™ i7-4790 CPU @ 3.6GHz processor and 16GB RAM.

As shown in Figure 3, this station includes 6 entering points, 5 leaving points, 12 platforms and 67 inbound and outbound routes. The traversal time of each route is set according to the length of the route and the average train speed in the yard. The shortest safety time interval permitted between two adjacent trains occupying the same platform $\varepsilon$ is set to 180 seconds.

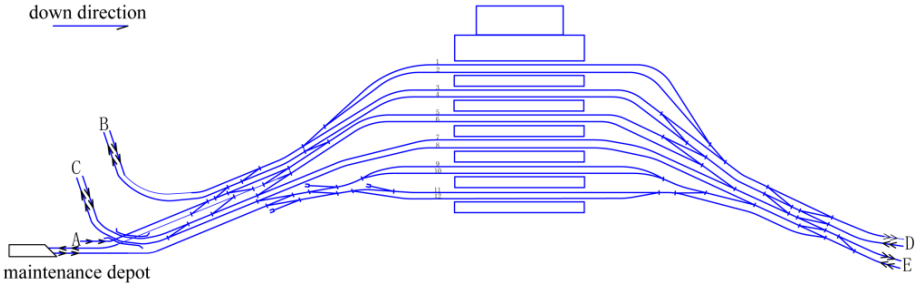


Figure 3: Layout of Zhengzhou East high-speed railway station.

Table 3 shows the detailed information for 84 trains. For each train, we record its entering point (EnP), its leaving point (LeP), its ideal arrival time (ArrTime), its departure time (DepTime) and its original allocated platform (Platform). The earliest train arrives at time 10:07:30 and the last train leaves station at 11:58:00. The entire considered time horizon is 2 hours.

Table 3: Train information.

| ID | EnP | LeP | ArrTime | DepTime | Platform | ID | EnP | LeP | ArrTime | DepTime | Platform |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 5 | 11 | 10:07:30 | 10:10:00 | 3 | 43 | 6 | 3 | 10:40:50 | 10:50:00 | 3 |
| 2 | 8 | 11 | 10:09:20 | 10:15:00 | 4 | 44 | 6 | 7 | 11:04:40 | 11:11:40 | 2 |
| 3 | 4 | 1 | 10:08:20 | 10:16:00 | 5 | 45 | 8 | 1 | 10:54:50 | 11:00:50 | 3 |
| 4 | 10 | 9 | 10:07:30 | 10:15:00 | 9 | 46 | 8 | 3 | 11:00:40 | 11:05:00 | 4 |
| 5 | 6 | 9 | 10:07:30 | 10:09:50 | 6 | 47 | 8 | 7 | 11:05:40 | 11:11:40 | 1 |
| 6 | 2 | 7 | 10:04:30 | 10:06:30 | 4 | 48 | 10 | 11 | 10:55:00 | 11:01:40 | 10 |
| 7 | 5 | 1 | 10:02:30 | 10:11:00 | 1 | 49 | 2 | 11 | 11:04:10 | 11:11:40 | 11 |
| 8 | 8 | 1 | 10:04:00 | 10:06:00 | 2 | 50 | 2 | 9 | 10:59:10 | 11:06:10 | 8 |
| 9 | 10 | 11 | 10:02:30 | 10:20:00 | 10 | 51 | 2 | 9 | 11:09:10 | 11:16:10 | 10 |
| 10 | 2 | 11 | 10:09:20 | 10:25:00 | 7 | 52 | 10 | 11 | 11:00:50 | 11:06:40 | 9 |
| 11 | 2 | 9 | 10:14:20 | 10:20:00 | 8 | 53 | 10 | 9 | 11:05:50 | 11:11:10 | 12 |
| 12 | 2 | 9 | 10:19:20 | 10:25:00 | 9 | 54 | 5 | 11 | 11:09:50 | 11:16:40 | 3 |
| 13 | 10 | 11 | 10:12:30 | 10:30:00 | 11 | 55 | 8 | 11 | 11:14:40 | 11:21:40 | 6 |
| 14 | 10 | 9 | 10:17:30 | 10:30:00 | 12 | 56 | 4 | 1 | 11:06:20 | 11:11:40 | 5 |
| 15 | 4 | 3 | 10:01:40 | 10:03:40 | 5 | 57 | 10 | 9 | 11:15:00 | 11:22:00 | 9 |
| 16 | 4 | 7 | 10:20:40 | 10:22:40 | 5 | 58 | 6 | 9 | 11:21:00 | 11:27:00 | 5 |
| 17 | 6 | 3 | 10:20:40 | 10:29:40 | 6 | 59 | 2 | 7 | 11:14:40 | 11:16:40 | 4 |
| 18 | 6 | 7 | 10:15:40 | 10:17:40 | 1 | 60 | 5 | 1 | 11:14:50 | 11:19:20 | 2 |
| 19 | 8 | 3 | 10:10:50 | 10:19:40 | 2 | 61 | 8 | 1 | 11:19:50 | 11:24:20 | 1 |
| 20 | 5 | 3 | 10:14:50 | 10:24:40 | 3 | 62 | 10 | 11 | 11:20:00 | 11:26:50 | 11 |
| 21 | 5 | 7 | 10:22:30 | 10:27:40 | 1 | 63 | 2 | 11 | 11:19:40 | 11:31:50 | 12 |
| 22 | 4 | 9 | 10:27:30 | 10:35:00 | 5 | 64 | 2 | 9 | 11:25:00 | 11:32:00 | 7 |
| 23 | 8 | 7 | 10:30:40 | 10:32:40 | 2 | 65 | 2 | 9 | 11:30:00 | 11:37:00 | 8 |
| 24 | 8 | 9 | 10:35:40 | 10:40:00 | 6 | 66 | 10 | 11 | 11:25:50 | 11:36:50 | 10 |
| 25 | 6 | 1 | 10:25:40 | 10:27:40 | 4 | 67 | 10 | 9 | 11:30:50 | 11:42:00 | 9 |
| 26 | 6 | 11 | 10:32:20 | 10:35:00 | 4 | 68 | 4 | 3 | 11:26:20 | 11:33:20 | 6 |
| 27 | 2 | 1 | 10:34:10 | 10:36:10 | 3 | 69 | 4 | 7 | 11:31:40 | 11:37:40 | 5 |
| 28 | 4 | 11 | 10:39:40 | 10:41:40 | 5 | 70 | 6 | 3 | 11:31:10 | 11:38:30 | 3 |
| 29 | 2 | 3 | 10:39:40 | 10:45:00 | 4 | 71 | 6 | 7 | 11:26:50 | 11:32:40 | 4 |
| 30 | 10 | 11 | 10:29:10 | 10:46:40 | 10 | 72 | 8 | 3 | 11:31:20 | 11:43:50 | 1 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 31 | 2 | 11 | 10:44:40 | 10:56:40 | 11 | 73 | 5 | 3 | 11:37:30 | 11:48:50 | 2 |
| 32 | 2 | 9 | 10:49:40 | 10:51:10 | 8 | 74 | 5 | 7 | 11:44:10 | 11:47:40 | 3 |
| 33 | 2 | 9 | 10:54:10 | 10:56:10 | 7 | 75 | 10 | 11 | 11:35:50 | 11:41:50 | 11 |
| 34 | 10 | 11 | 10:37:30 | 10:51:40 | 9 | 76 | 2 | 11 | 11:37:10 | 11:46:50 | 12 |
| 35 | 10 | 9 | 10:50:00 | 11:01:10 | 12 | 77 | 2 | 9 | 11:42:10 | 11:47:00 | 10 |
| 36 | 4 | 1 | 10:46:20 | 10:50:50 | 5 | 78 | 2 | 9 | 11:47:10 | 11:52:00 | 9 |
| 37 | 4 | 3 | 10:51:20 | 10:57:10 | 6 | 79 | 10 | 11 | 11:49:10 | 11:51:50 | 11 |
| 38 | 4 | 7 | 10:56:20 | 11:01:40 | 5 | 80 | 2 | 9 | 11:52:10 | 11:57:30 | 10 |
| 39 | 5 | 1 | 10:40:00 | 10:45:50 | 2 | 81 | 10 | 11 | 11:54:10 | 11:56:50 | 12 |
| 40 | 5 | 3 | 10:32:30 | 10:40:00 | 1 | 82 | 4 | 1 | 11:42:30 | 11:48:20 | 5 |
| 41 | 5 | 7 | 10:49:50 | 10:56:40 | 1 | 83 | 5 | 1 | 11:53:40 | 11:58:00 | 1 |
| 42 | 6 | 1 | 10:49:40 | 10:55:50 | 4 | 84 | 4 | 3 | 11:47:30 | 11:52:10 | 6 |

The following disruption situation is considered: (1) train 6 is two minutes behind schedule; (2) platform 5 is unavailable from 10:30:00 to 10:40:00; (3) train 44 is five minutes behind schedule. In this case, the total number of train paths is 379. The model of the overall problem has 1,433,882 constraints and 35,954 variables, which takes 3600s to obtain a solution with 58.94% duality gap by using the CPLEX solver. By using the iterative algorithm, the computational time of this instance is less than two seconds. The number of iterations is two. The minimum overall deviation from the ideal planed schedule of each train is 320 seconds.

The instance size is similar to the medium sized railway station Arnhem presented by Zwaneveld et al. (2001). Arnhem has 16 platform and is visited by about 40 trains per hour. But since the problem considered by Zwaneveld et al. (2001) is at the strategic level and their computer computing power is different from ours, we cannot compare the computational efficiency of our approach with their solving methods. However, the results still clearly demonstrate the great potential of our iterative algorithm.

Figure 4 shows the original platform allocation plan. Figure 5 shows the adjusted platform allocation. In these two figures, each rectangle represents a train occupying a platform, the length of each rectangle represents the duration of occupying the platform and the number to the right of each rectangle indicates the train ID. The rectangle with light colour implies that the corresponding train's arrival and departure time, its platform, inbound and outbound routes are all not changed; while the rectangle with dark colour implies that the corresponding train is rescheduled or reassigned a different platform, inbound route or outbound route. Table 4 shows the information of trains which is rescheduled or reassigned a different path.

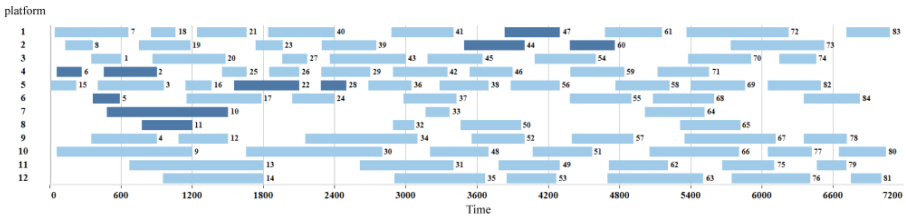


Figure 4: The original platform allocation plan.

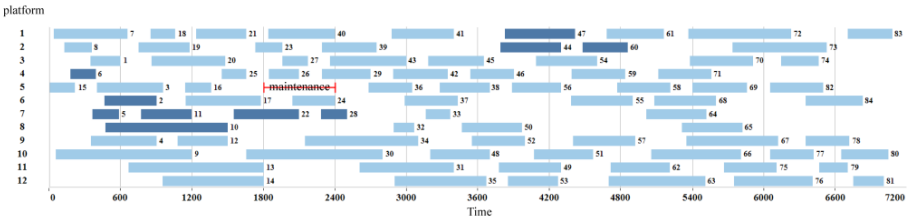


Figure 5: The adjusted platform allocation plan.

Table 4: Information of trains which is rescheduled or reassigned a different path.

| ID | Original plan | | | Adjusted plan | | |
|---|---|---|---|---|---|---|
| | ArrTime | DepTime | Platform | ArrTime | DepTime | Platform |
| 2 | 10:09:20 | 10:15:00 | 4 | 10:09:20 | 10:15:00 | 6 |
| 5 | 10:07:30 | 10:09:50 | 6 | 10:07:30 | 10:09:50 | 7 |
| 6 | 10:02:30 | 10:04:30 | 4 | 10:04:30 | 10:06:30 | 4 |
| 10 | 10:09:20 | 10:25:00 | 7 | 10:09:20 | 10:25:00 | 8 |
| 11 | 10:14:20 | 10:20:00 | 8 | 10:14:20 | 10:20:00 | 7 |
| 22 | 10:27:30 | 10:35:00 | 5 | 10:27:30 | 10:35:00 | 7 |
| 28 | 10:39:40 | 10:41:40 | 5 | 10:39:40 | 10:41:40 | 7 |
| 44 | 10:59:40 | 11:06:40 | 2 | 11:04:40 | 11:11:40 | 2 |
| 47 | 11:05:40 | 11:11:40 | 1 | 11:05:40 | 11:13:40 | 1 |
| 60 | 11:14:50 | 11:19:20 | 2 | 11:16:30 | 11:21:00 | 2 |

## 6 Conclusions

In this paper we studied the real-time train platforming and routing problem at a busy complex high-speed railway station in a disrupted situation. A bi-objective mixed integer linear programming model was formulated to solve the problem, which is also a universal model for the route-lock sectional/integral release interlocking system. An iterative algorithm is designed to solve the MILP model efficiently. The results of real-world experiment based on the Zhengzhou East high-speed railway station show the proposed model and solution approach have great potential to be applied in the real-world operations.

Our future research will focus on the following significant aspects:

1. This paper assumes that all trains travel at the same constant speed regardless of route choice. The speed profile for each train can be taken into consideration in order to meet the operational requirements.
2. To the feasibility of the solution, this paper enforces that all trains are not permitted to arrive earlier than its ideal arrival time. In fact, trains are permitted to arrive a few minutes earlier without violating the condition of sections. Thus, the proposed model and algorithm can be extended to the railway network.
3. At the tactical level, based on the fixed arrival and departure times, reasonable adjustment to the platform allocation plan and the route assignment plan can increase buffer time. Buffer time can be used to absorb train delay to some extent. Thus, the robustness of platform allocation and route assignment plan can be another research direction.

## Acknowledgements

The research was supported by National Key Research and Development Program of China (No. 2017YFB1200700-1) and National Natural Science Foundation of China (No. U1834209).

## References

Boccia, M., Mannino, C., Vasilyev, I., 2013. "The dispatching problem on multitrack territories: Heuristic approaches based on mixed integer linear programming", *Networks*, vol. 62, pp. 315-326.

Cacchiani, V., Huisman, D., Kidd, M., et al., 2014. "An overview of recovery models and algorithms for real-time railway rescheduling", *Transportation Research Part B: Methodological*, vol. 63, pp. 15-37.

Caimi, G., Fuchsberger, M., Laumanns, M., et al., 2012. "A model predictive control approach for discrete-time rescheduling in complex central railway station areas", *Computers & Operations Research*, vol. 39, pp. 2578–2593.

Caprara, A., Galli, L., Toth, P., 2011. "Solution of the train platforming problem", *Transportation Science*, vol. 45, pp. 246–257.

Carey, M., Carville, S., 2003. "Scheduling and platforming trains at busy complex stations", *Transportation Research Part A: Policy and Practice*, vol. 37, pp. 195–224.

D'ariano, A., Pacciarelli, D., Pranzo, M., 2007. "A branch and bound algorithm for scheduling trains in a railway network", *European Journal of Operational Research*, vol. 183, pp. 643-657.

D'Ariano, A., Corman, F., Pacciarelli, D., et al., 2008. "Reordering and local rerouting strategies to manage train traffic in real time", *Transportation science*, vol. 42, pp. 405-419.

Dollevoet, T., Huisman, D., Kroon, L., et al., 2014. "Delay management including capacities of stations", *Transportation Science*, vol. 49, pp. 185-203.

Lamorgese, L., Mannino, C., 2015. "An exact decomposition approach for the real-time train dispatching problem", *Operations Research*, vol. 63, pp. 48-64.

Lusby, R. M., Larsen, J., Ehrgott, M., et al., 2011. "Railway track allocation: models and methods", *OR spectrum*, vol. 33, pp. 843-883.

Lusby, R. M., Larsen, J., Ryan, D., et al., 2011. "Routing trains through railway junctions: a new set-packing approach", *Transportation Science*, vol. 45, pp. 228-245.

Lusby, R. M., Larsen, J., Ehrgott, M., et al., 2013. "A set packing inspired method for real-time junction train routing", *Computers & Operations Research*, vol. 40, pp. 713-724.

Meng, L., Zhou, X., 2014. "Simultaneous train rerouting and rescheduling on an N-track network: A model reformulation with network-based cumulative flow variables", *Transportation Research Part B: Methodological*, vol. 67, pp. 208-234.

Pellegrini, P., Marlière, G., Pesenti, R., et al., 2015. "RECIFE-MILP: An effective MILP-based heuristic for the real-time railway traffic management problem", *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, pp. 2609-2619.

Pellegrini, P., Pesenti, R., Rodriguez, J., 2019. "Efficient train re-routing and rescheduling: Valid inequalities and reformulation of RECIFE-MILP", *Transportation Research Part B: Methodological*, vol. 120, pp. 33-48.

Rodriguez, J., 2007. “A constraint programming model for real-time train scheduling at junctions”, *Transportation Research Part B: Methodological*, vol. 41, pp. 231-245.

Rodriguez, J., Kermad, L., 1998. “Constraint programming for real-time train circulation management problems in railway nodes”, *WIT Transactions on The Built Environment*, vol. 37.

Sama, M., Pellegrini, P., D’Ariano, A., et al., 2016. “Ant colony optimization for the real-time train routing selection problem”, *Transportation Research Part B: Methodological*, vol. 85, pp. 89-108.

Sels, P., Vansteenwegen, P., Dewilde, T., et al., 2014. “The train platforming problem: The infrastructure management company perspective”, *Transportation Research Part B: Methodological*, vol. 61, pp. 55-72.

Zwaneveld, P.J., Kroon, L.G., Van Hoesel, S.P.M., 2001. “Routing trains through a railway station based on a node packing model”, *European Journal of Operational Research*, vol. 128, pp. 14-33.

Zwaneveld, P.J., Kroon, L.G., Romeijn, H.E., et al., 1996. “Routing trains through railway stations: Model formulation and algorithms”, *Transportation science*, vol. 30, pp. 181-194.

Wu, Q., Hao, J., 2015. “A review on algorithms for maximum clique problems”, *European Journal of operational research*, vol. 242, pp. 693-709.