Netw Spat Econ (2017) 17:1–39
DOI 10.1007/s11067-015-9316-4

Branch-and-Bound-Based Local Search Heuristics for Train Timetabling on Single-Track Railway Network

Ampol Karoonsoontawong[1] · Apisak Taptana[1]

Published online: 19 December 2015

Abstract The modified formulation and two branch-and-bound-based local search heuristic algorithms for train timetabling in single-track railway network in the planning application are proposed. The original local search heuristic is modified such that when a neighbor of a currently tested resolved conflict for improvement is evaluated, the depth-first search branch-and-bound algorithm is employed with two branching rules: least-lower-bound and least-delay-time. The detailed implementation of the proposed heuristic algorithms are described, including the neighborhood definitions of overtaking and crossing conflicts, the procedure to detect overtaking and crossing conflicts, and a recursive procedure for the depth-first search branch-and-bound algorithm. The proposed heuristic algorithms, the original heuristic, the equivalent manual solution method and the exact solution method are compared using a toy problem and four problems (26-train, 50-train, 76-train and 108-train) in the Thailand Southern line railway network, which consisted of 266 single-track segments, 15 double-track segments, 282 stations/sidings, and total distance of 1577 km. The proposed heuristic algorithm with the least-lower-bound branching rule outperforms the other heuristics in terms of the solution quality with up to 2.827 % improvement over the equivalent manual solution method and less than 8 % optimality gap. The proposed heuristic algorithms require longer time to terminate than the original heuristic. The proposed heuristic algorithm with the least-lower-bound branching rule converges faster than that with the least-delay-time branching rule in all test problems. Based on the empirical results, the proposed heuristic algorithms are solvable in polynomial time. Furthermore, the proposed heuristic algorithm with the least-lower-bound branching rule is enhanced by embedding a uniform sampling strategy, and it is found that the total CPU time can be saved by about 50 % with marginally worse solution quality.

Keywords Train timetabling · Local search heuristic · Branch and bound algorithm

✉ Ampol Karoonsoontawong
ampolk@gmail.com

[1] Department of Civil Engineering, Faculty of Engineering, King Mongkut's University of Technology Thonburi, 126 Pracha Utid Rd., BangmodThung KhruBangkok 10140, Thailand

1 Introduction

Rail transportation is an efficient and economic transportation mode for passengers and goods. Train schedules are important in the decision-making process of passengers and freight carriers. Train schedules are the basis for locomotive and crew scheduling, and the quality of train schedules affects the investments and operating costs. Essentially, train schedule optimization can help the railway industries increase revenues and decrease costs.

In our view, the contributions of our work are threefold. First, the formulation for a single-track line in Higgins et al. (1996a) is extended to accommodate the railway network, and implemented in IBM ILOG CPLEX Optimization Studio IDE Version 12.6 in order to determine the exact solution as well as the best lower bound. The best lower bound can be used to calculate the optimality gap of heuristic solutions. Second, the local search heuristic in Higgins et al. (1997) is modified so that the depth-first-search branch-and-bound algorithm is used to generate the initial solution and embedded in the local search heuristic algorithm. Two branching rules (least-lower-bound and least-delay-time) are considered in the branch-and-bound-based local search heuristic, which results in two proposed heuristics. The neighborhoods, which account for the segment type (single-track or double-track), for the resolved overtaking and crossing conflicts are explicitly defined. Furthermore, we demonstrate that the total computational time of the proposed heuristics can significantly be improved by incorporating uniform sampling strategies. Third, a real-world problem, i.e., the Southern Line Railway Network in Thailand, which has never been used in the literature, is presented as a benchmark problem.

In the next section, the literature review is provided, followed by the modified formulation and the description of the branch-and-bound-based local search heuristics. The computational results of the real-world case study in Thailand are then shown.

2 Literature Review

The comprehensive surveys of the train-scheduling problem and its inherent connection to other railroad-routing and -scheduling problems are found in Cordeau et al. (1998), Newman et al. (2002) and Meng et al. (2015). The literature on train scheduling can be classified into two categories (Zhou and Zhong 2005): planning and real-time. In planning applications, train scheduling generates train timetables to minimize the overall operational costs and satisfy the passenger and freight traffic demands. The recent advancements of traffic demand models can be found in Karoonsoontawong and Lin (2015), Liu and Huang (2015) and Xiao and Lo (2014), and those of freight transportation modeling can be found in Han et al. (2015), Karoonsoontawong (2015), Karoonsoontawong and Heebkhoksung (2015) and Karoonsoontawong and Unnikrishnan (2013). The train scheduling in planning applications can further be classified into two classes: line planning and schedule generation. In line-planning applications, the train routes, frequencies, and scheduled stop times are determined. Assad (1980) provided a comprehensive review on the line-planning applications. Bussieck et al. (1997) proposed an integer program for line planning in a railroad network. Chang et al. (2000) proposed a multi-objective linear programming formulation to determine the stop and frequency schedules. Li et al. (2011) developed a local search algorithm to optimize train routing on railway network. In schedule generation, the arrival and departure times of each train at passing stations or sidings are determined. In the pioneering paper of Szpigel (1973), the problem of scheduling trains on a single-track line to minimize the total travel time was modeled as a job shop scheduling problem and solved using a branch-and-bound algorithm in Greenberg (1968). The exact solution methods were proposed in the following works: the operational train scheduling problem that minimized the deviation from the planned schedules (Kraay and Harker 1995), lower-bound estimation procedure for the branch-and-bound algorithm (Higgins et al. 1996a), Lagrangian relaxation methods (Brannlund et al. 1998), and branch-and-bound algorithm with effective dominance rules to generate Pareto solutions for the bicriteria train-scheduling problem for high-speed passenger railroad planning (Zhou and Zhong 2005). Our work belongs to the schedule generation category in the planning application. In real-time applications, train scheduling adjusts the daily and hourly train operation timetables to enhance the on-time performance and reliability. Two features are discussed in the literature: efficient heuristic methods (e.g., Adenso-Diaz et al. 1999; Sahin 1999; D'Ariano et al. 2007; D'Ariano and Pranzo 2009; Albrecht 2009) and train schedule reliability (e.g., Kraay and Harker 1995; Higgins and Kozan 1998; Higgins et al. 1996b). Higgins and Kozan (1998), Higgins et al. (1996b), Ma et al. (2002), Zhou and Zhong (2005) and Khan and Zhou (2010) examined the multi-criteria train-scheduling problem.

The train-scheduling problem is non-deterministic polynomial-time hard (NP-hard) (Cai et al. 1998; Caprara et al. 2002); thus, a heuristic approach is typically adopted to generate feasible and suboptimal solutions in large-scale and complex instances. The literature on the mathematical-programming-based solution methods includes the branch-and-bound algorithms (Szpigel 1973; Jovanovic and Harker 1991; Higgins et al. 1996a) and decomposition algorithms (Caprara et al. 2002; Carey 1994a, b; Carey and Lockwood 1995; Brannlund et al. 1998; Zhou and Zhong 2005, 2007). The literature on heuristic solution methods may be classified into two categories: priority rules and complex search procedure. The priority-rule-based heuristics consider the total delay to minimize the travel time (Petersen et al. 1986; Kraay and Harker 1995), schedule deviation and tightness to minimize the deviation (Chen and Harker 1990), train priorities (Petersen et al. 1986) and the number of transported passengers (Adenso-Diaz et al. 1999). The complex search procedures are backtracking search (Adenso-Diaz et al. 1999), look-ahead search (Sahin 1999), and metaheuristic algorithms (Higgins et al. 1996a, 1997; Jong et al. 2013). Our proposed solution method belongs to the complex-search-procedure heuristic category.

Meng and Zhou (2014) systematically classified the studies of train timetabling and train dispatching according to four criteria: scheduling strategy, scheduling measure, formulation type and solution algorithm. First, simultaneous scheduling strategy determines both train route and arrival/departure times of trains, whereas sequential scheduling strategy first specifies train routes and then determine the arrival/departure times of each train. Second, scheduling measure includes time, order, track, route, re-time, re-order, re-track (or local re-route) and re-route (or global re-route). Third, formulation type includes integer programming, mixed integer programming (MIP) and simulation-based model. Solution algorithm includes branch-and-bound (B&B), branch-and-cut-and-price (BCP), alternative graphs (AG), Lagrangian relaxation (LR), heuristics (H), dynamic programming (DP), local search (LS), practical rules (PR), bisection method (BS), column generation (CG) and commercial solver (CS). Table 1 shows the systematic classification of train scheduling literature according to problem type (train time tabling or train dispatching), type of objective (single or multiple) and the above-mentioned four criteria.

As shown in Table 1, our work belongs to the class of train time tabling, sequential scheduling strategy, schedule measures of time and order, single objective, mixed integer program and the solution algorithm of branch-and-bound-based local search. To our knowledge, our work is the first to embed the branch-and-bound algorithm into the local search heuristic. Our proposed work is based on Higgins et al. (1996a, 1997). Higgins et al. (1996a) developed a mathematical model to optimize train schedules on single-line rail corridors, and the depth-first-search branch-and-bound algorithm. At a node of the branch-and-bound tree, which corresponds to an unresolved train schedule, the first conflict in time with two involved trains is selected, and two branches are possible from this node: (i) a train involved in the conflict is delayed, and (ii) the other train is delayed. For each possible branch, the lower bound is determined, and the branching rule is to select the branch with the minimal lower bound (i.e., the least-lower-bound branching rule). Higgins et al. (1997) proposed the local search heuristic that used the depth-first-search branch-and-bound algorithm to determine the initial solution. The local search heuristic repeatedly shifts the position of a resolved conflict to its neighbors, and any new conflicts resulted from these moves are resolved to the nearest sidings (i.e., the least-delay-time branching rule).

3 Modified Formulation

Higgins et al. (1996a) and Zhou and Zhong (2007) proposed a mixed integer linear program for train scheduling on a single-track rail line. We modify the formulation in Higgins et al. (1996a) for a railway network. The model assumes the followings:

- The railway network is divided into segments, which are separated with sidings or stations.
- Trains can follow one another on a track segment with the minimum distance headway equal to the track length; i.e., only one train is permitted on any track segment.
- For double-track segments, one lane is allocated for inbound trains, and the other lane is allocated for outbound trains.
- Overtaking can occur at any siding or double-track rail segments; crossing can occur at any single-track rail segments.
- Scheduled stops are permitted at intermediate station/siding for any train (Assume that the length of siding track at any station/siding is longer than the length of any train.)
- Each train traverses on a single pre-specified path.
- There is not a segment with more than two tracks.
- There are two tracks at sidings/stations.

The sets, parameters, decision variables, objective function and constraints are described as follows.

Table 1 Classification of train scheduling literature (After Meng and Zhou 2014)

Publication	Scheduling strategy	Scheduling measure	Formulation type	Single- /Multiple- Objective	Solution algorithm
Train dispatching literature					
Adenso-Diaz et al. (1999)	Sequential	Re-Time, Re-Order	Mixed Integer	Single	H
Tornquist and Persson (2007)	Sequential, Simultaneous	Re-Time, Re-Order, Re-track, Re-route	Mixed Integer	Single	H
D'Ariano et al. (2007)	Sequential	Re-Track, Re-Order	Mixed Integer	Single	B&B, H
D'Ariano et al. (2008)	Sequential	Re-Time, Re-Order, Re-Track, Re-Route	Mixed Integer	Single	B&B, AG, LS, PR
Corman et al. (2010)	Sequential	Re-Time, Re-Order, Re-Track, Re-Route	Mixed Integer	Single	B&B, H
Corman et al. (2011)	Sequential	Re-Time, Re-Order	Mixed Integer	Single, Multiple	B&B, H
Caimi et al. (2012)	Sequential	Re-Time, Re-Order, Re-Track	Integer	Single	CS
Meng and Zhou (2014)	Simultaneous	Re-Time, Re-Order, Re-Track, Re-Route	Integer	Single	LR, H
Train time tabling literature					
Higgins et al. (1996a)	Sequential	Time, Order	Mixed Integer	Single	B&B
Higgins et al. (1997)	Sequential	Time, Order	Mixed Integer	Single	LS
Brannlund et al. (1998)	Sequential	Time, Order	Integer	Single	LR, H
Caprara et al. (2002)	Sequential	Time, Order	Integer	Single	LR, H
Dorfman and Medanic (2004)	Sequential	Time, Order	Simulation	Multiple	H
Carey and Crawford (2007)	Sequential	Time, Order, Track	–	Single	H
Cacchiani et al. (2008)	Sequential	Time, Order	Integer	Single	CG, H, BCP
Lee and Chen (2009)	Sequential	Time, Order, Track, Route	Mixed Integer	Single	H

Table 1 (continued)

Publication	Scheduling strategy	Scheduling measure	Formulation type	Single- /Multiple- Objective	Solution algorithm
Cacchiani et al. (2010a)	Sequential	Time, Order	Integer	Single	DP, H
Cacchiani et al. (2010b)	Simultaneous	Time, Order, Route	Integer	Single	DP, H, LR
Castillo et al. (2011)	Sequential	Time, Order	Mixed Integer	Single	BS
Mu and Dessouky (2011)	Sequential, Simultaneous	Time, Order, Track	Mixed Integer	Single	H
Harrod (2011)	Simultaneous	Time, Order, Track	Integer	Single	CS
Karoonsoontawong and Taptana (This Paper)	Sequential	Time, Order	Mixed Integer	Single	LS, B&B

AG Alternative Graph, *BCP* Branch-and-Cut-and-Price Algorithm, *BS* Bisection Method, *B&B* Branch-and-Bound Algorithm, *CG* Column Generation, *CS* Commercial Solver, *DP* Dynamic Programming, *H* Heuristic, *LR* Lagrangian Relaxation, *LS* Local Search, *PR* Practical Rules

3.1 Sets

I set of trains

P set of rail segments

$P^{same\_d}(i,j)$ set of common rail segments for trains i and j, which traverse in the same direction

$P(i)$ ordered set of rail segments traversed by train i (i.e., the route for train i)

Q set of stations/sidings (In this paper, "station" and "siding" are used interchangeably.)

$Q(i)$ ordered set of stations/sidings traversed by train i (i.e., the route for train i)

P_1 set of single-track segments

$P_1^{opp\_d}(i,j)$ set of common single-track segments for trains i and j, which traverse in opposite directions

P_2 set of double-track segments

D set of segment directions (inbound and outbound)

3.2 Parameters

$q_1(p,d)$ beginning siding/station of segment p in direction d

$q_2(p,d)$ terminal siding/station of segment p in direction d

$d_{i,p}$ direction that train $i \in I$ traverses segment $p \in P(i)$

$h_{1,p}^{i,j}$ minimum safety headway between trains $i,j \in I$ traversing in the same direction on $p \in P$

$h_{2,p}^{i,j}$ minimum safety headway between trains $i,j \in I$ traveling in opposite directions on $p \in P_1$

l_p length of segment $p \in P$

Y_{dO}^{i} earliest departure time of train $i \in I$ from its origin station

$\underline{v}_p^i$ minimum allowable average velocity of train $i \in I$ on segment $p \in P(i)$

$\overline{v}_p^i$ maximum achievable average velocity of train $i \in I$ on segment $p \in P(i)$ ($\underline{v}_p^i$ and $\overline{v}_p^i$ depend on the physical characteristics of the track segment and the train.)

W_i priority weight of train $i \in I$ (a higher priority corresponds to a larger priority weight)

S_q^i scheduled stop time for train $i \in I$ at station $q \in Q(i)$

M sufficiently large constant to satisfy the crossing and overtaking constraints

3.3 Decision Variables

A_{ijp} 1 if train $i \in I$ traverses track segment $p \in P^{same\_d}(i,j)$ before train $j \in I$ when trains i and j traverse track segment p in the same direction; 0 otherwise.

B_{ijp} 1 if train $i \in I$ traverses track segment $p \in P_1^{opp\_d}(i,j)$ before train $j \in I$ when trains i and j traverse track segment p in opposite directions; 0 otherwise.
$X^i_{a,q}$ arrival time of train $i \in I$ at station/siding $q \in Q(i)$
$X^i_{d,q}$ departure time of train $i \in I$ from station/siding $q \in Q(i)$
X^i_{dO} departure time of train $i \in I$ from its origin station
X^i_{aD} arrival time of train $i \in I$ at its destination station
z objective function value

3.4 Objective Function

$$\min z = \sum_{i \in I} W_i \left(X^i_{aD} - Y^i_{dO} \right) \tag{1.1}$$

3.5 Constraints

$$M \cdot A_{ijp} + X^i_{d,q_1(p,d_{i,p})} \geq X^j_{a,q_2(p,d_{j,p})} + h^{i,j}_{1,p} \quad \forall i \neq j \in I, p \in P^{same\_d}(i,j) \tag{1.2}$$

$$M \cdot \left(1 - A_{ijp}\right) + X^j_{d,q_1(p,d_{j,p})} \geq X^i_{a,q_2(p,d_{i,p})} + h^{i,j}_{1,p} \quad \forall i \neq j \in I, p \in P^{same\_d}(i,j) \tag{1.3}$$

$$M \cdot B_{ijp} + X^i_{d,q_1(p,d_{i,p})} \geq X^j_{a,q_2(p,d_{j,p})} + h^{i,j}_{2,p} \quad \forall i \neq j \in I, p \in P_1^{opp\_d}(i,j) \tag{1.4}$$

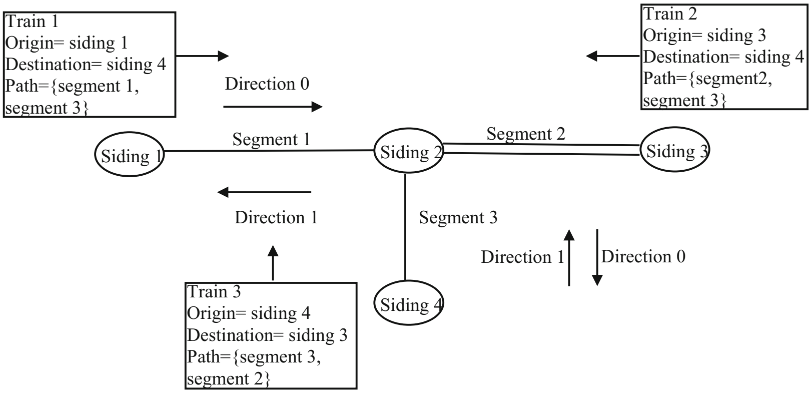
$$M \cdot \left(1 - B_{ijp}\right) + X^j_{d,q_1(p,d_{j,p})} \geq X^i_{a,q_2(p,d_{i,p})} + h^{i,j}_{2,p} \quad \forall i \neq j \in I, p \in P_1^{opp\_d}(i,j) \tag{1.5}$$

$$\frac{l_p}{\bar{v}^i_p} \leq X^i_{a,q_2(p,d_{i,p})} - X^i_{d,q_1(p,d_{i,p})} \leq \frac{l_p}{\underline{v}^i_p} \quad \forall i \in I, p \in P(i) \tag{1.6}$$

$$X^i_{dO} \geq Y^i_{dO} \quad \forall i \in I \tag{1.7}$$

$$X^i_{a,q} + S^i_q \leq X^i_{d,q} \quad \forall i \in I; q \in Q(i) \tag{1.8}$$

The objective (1.1) minimizes the weighted sum of the train travel times. Constraint sets (1.2)–(1.3) enforce that for any two trains i and j traversing the same segment p in the same direction, the value of A_{ijp} is equal to zero if and only if train j traverses segment p before train i, and train j must leave the segment p for the period of $h^{i,j}_{1,p}$ before train i can enter it; i.e., $X^i_{d,q_1(p,d_{i,p})} \geq X^j_{a,q_2(p,d_{j,p})} + h^{i,j}_{1,p}$. Constraint sets (1.2)-(1.3) also enforce that $A_{ijp} = 1$ if and only if train traverses segment p before train j. Constraint sets (1.4)–(1.5) ensure that for any two trains i and j traversing the same single-track segment p in opposite directions, the value of B_{ijp} is equal to one if and only if train i traverses segment p before train j with the time headway greater than or equal to the minimum safety headway; i.e., $X^j_{d,q_1(p,d_{j,p})} \geq X^i_{a,q_2(p,d_{i,p})} + h^{i,j}_{2,p}$.



$I=\{\text{train 1, train 2, train 3}\}$	$P_1=\{\text{segment 1, segment 3}\}$
$P=\{\text{segment 1, segment 2, segment 3}\}$	$P_2=\{\text{segment 2 }\}$
$Q=\{\text{siding 1, siding 2, siding 3, siding 4}\}$	$P_1^{opp\_d}(\text{train 2}, \text{train 3})=\{\text{segment 3}\}$
$P^{same\_d}(\text{train 1}, \text{train 2})=\{\text{segment 3}\}$	$D=\{\text{direction 0, direction 1 }\}$
$P(\text{train 1})=\{\text{segment 1, segment 3}\}$	$q_1(\text{segment1, direction 1})=\text{siding 2}$
$P(\text{train 2})=\{\text{segment 2, segment 3}\}$	$q_2(\text{segment1, direction 1})=\text{siding 1}$
$P(\text{train 3})=\{\text{segment 3, segment 2}\}$	$d_{\text{train 1,segment 3}}=\text{direction 0}$
$Q(\text{train 1})=\{\text{siding 1, siding 2, siding 4}\}$	$d_{\text{train 3,segment 3}}=\text{direction 1}$
$Q(\text{train 2})=\{\text{siding 3, siding 2, siding 4}\}$	
$Q(\text{train 3})=\{\text{siding 4, siding 2, siding 3}\}$	

Fig. 1 Rail network example

Constraint sets (1.4)–(1.5) also ensures that B_{ijp} is equal to zero if and only if train j traverses segment p before train i with the time headway greater than or equal to the minimum safety headway. Constraint set (1.6) ensures that the train travel times on any rail segment are within the corresponding upper and lower limits. Constraint set (1.7) allows the train departure times from their origin stations to be greater than or equal to their earliest departure times. Constraint set (1.8) states that a train departs a station/siding after it arrives at this station and stops there for at least the scheduled stop time.

Figure 1 illustrates a rail network example to clarify certain notations. Given the known routes for trains i and j (i.e., known $P(i)$ and $P(i)$), the set $P^{same\_d}(i,j)$ includes all common rail segments (single-track or double-track) for trains i and j, where an overtaking conflict may occur. The set $P_1^{opp\_d}(i,j)$ includes all common single-track rail segments for trains i and j where a crossing conflict may occur. Then, the proposed formulation can capture possible crossing and overtaking conflicts in the railway network. The objective function compresses the timetable to maximize the effective capacity of the infrastructure, but it may lead to a less robust timetable. One may obtain a more robust timetable by using the underestimated average train velocities, so that the resulted train timetable is more flexible.

Specifically, the modifications to the original formulation are the addition of the following notations: $P(i)$, $Q(i)$, $P^{same\_d}(i,j)$, $P_1^{opp\_d}(i,j)$, $d_{i,p}$, $q_1(p,d)$, and $q_2(p,d)$. $P(i)$ and $Q(i)$ are used to store the path of each train i. $P^{same\_d}(i,j)$, $P_1^{opp\_d}(i,j)$, $d_{i,p}$, $q_1(p,d)$ and $q_2(p,d)$ are used to efficiently define the constraint sets (1.2)–(1.6). The modifications also change the following notations: $h_{1,p}^{i,j}$, $h_{2,p}^{i,j}$, A_{ijp} and B_{ijp}. In the original formulation, the headway parameters (h_p) depend on only the rail segment p.

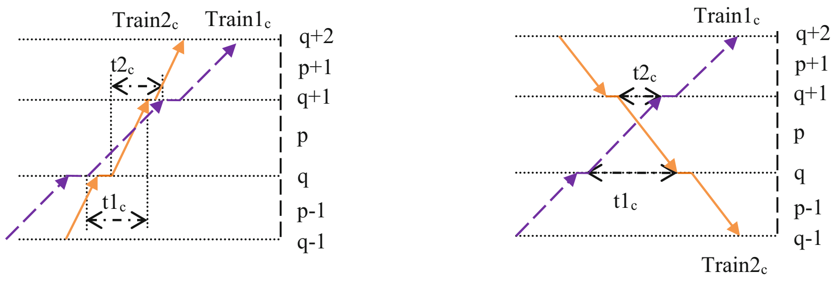
```
$Determine\_Schedule(\{S_q^i\},\{ES_q^i\},\{v_p^i\},\{X_{dO}^i\})$
For each train $i \in I$
    For each segment $p \in P(i)$
        $X^i_{a,q_2(p,d_{i,p})} = X^i_{d,q_1(p,d_{i,p})} + \frac{l_p}{v_p^i}$
        $X^i_{d,q_2(p,d_{i,p})} = X^i_{a,q_2(p,d_{i,p})} + S_q^i + ES_q^i$
Return $z = \sum_{i \in I} W_i (X_{aD}^i - Y_{dO}^i)$
```

Fig. 2 *Determine_Schedule* module

In the modified formulation, the headway parameters ($h_{1,p}^{i,j}$ and $h_{2,p}^{i,j}$) can vary and depend on the train pairs and train directions. In the original formulation, the binary decision variables A_{ijp} (for inbound trains i and j), B_{ijp} (for inbound train i and outbound train j) and C_{ijp} (for outbound trains i and j) are used to determine which train first traverses segment p. In the modified formulation, A_{ijp} is used to determine which train traverses first when trains i and j traverse in the same direction on segment p, and B_{ijp} is used when trains i and j traverse in opposite directions on segment p; therefore, C_{ijp} is eliminated. Lastly, the modified formulation guarantees that only a train can traverse a track segment in the same direction, whereas in the original formulation, more than one train can traverse the same segment in the same direction.

4 Branch-and-Bound-Based Local Search Heuristics

Higgins et al. (1997) proposed a local search heuristic, which is a basis of the proposed branch-and-bound-based local search heuristics in this paper.



a) Delay Times $t1$ and $t2$ for Resolving An Overtaking Conflict

b) Delay Times $t1$ and $t2$ for Resolving A Crossing Conflict

Note: p-1 = the immediately preceding segment to segment p
p+1= the immediately successive segment to segment p
q+1 = the immediately successive siding to siding q
q+2 = the immediately successive siding to siding q+1

Fig. 3 Delay times $t1$ and $t2$ for resolving overtaking conflict and crossing conflict

```
Find_Conflicts({ $X^{i}_{a,q}$ },{ $X^{i}_{d,q}$ }, num_resolved_c)
$c = num\_resolved\_c$
For each train $i_1 \in I$ ,
    For each train $i_2 \in I, i_1 \neq i_2$ ,
        For each segment $p_1 \in P(i_1)$
            For each segment $p_2 \in P(i_2)$
                If (segments $p_1$ and $p_2$ are the same; i.e. $p_1 = p_2 = \hat{p}$ )
                    If (the directions of trains $i_1$ and $i_2$ on this segment are the same; i.e., $d_{i_1,p_1} = d_{i_2,p_2} = \hat{d}$ )
                        If (train $i_2$ departs from the beginning station/siding of segment $\hat{p}$ after train
                        $i_1$ does within insufficient headway ; i.e. $X^{i_2}_{d,q_1(\hat{p},\hat{d})} \geq X^{i_1}_{d,q_1(\hat{p},\hat{d})}$ and
                        $X^{i_2}_{d,q_1(\hat{p},\hat{d})} < X^{i_1}_{a,q_2(\hat{p},\hat{d})} + safety\_h^{i_1,i_2}_{1,\hat{p}}$ )
                        {
                            $train1_c = i_1$ ; $train2_c = i_2$ ; $type_c = 1$; $time_c = X^{i_1}_{d,q_1(\hat{p},\hat{d})}$ ;
                            $t1_c = X^{i_2}_{a,q_2(\hat{p},\hat{d})} - X^{i_1}_{d,q_1(\hat{p},\hat{d})}$ ; $t2_c = X^{i_1}_{a,q_2(\hat{p},\hat{d})} - X^{i_2}_{d,q_1(\hat{p},\hat{d})}$ ;
                            $resolved_c = 0$
                            $d\_train1_c = d\_train2_c = \hat{d}$ ; $segment_c = \hat{p}$ ; $c = c+1$
                        }
                        Else if ((the directions of trains $i_1$ and $i_2$ on this segment are opposite; i.e., $d_{i_1,p_1} \neq d_{i_2,p_2}$ )
                            If (segment $\hat{p}$ is single-track; i.e. $num\_line_{\hat{p}} = 1$)
                                If ( $X^{i_2}_{d,q_1(\hat{p},d_{i_2,p_2})} < X^{i_1}_{a,q_2(\hat{p},d_{i_1,p_1})} + safety\_h^{i_1,i_2}_{2,\hat{p}}$ and
                                $X^{i_1}_{d,q_1(\hat{p},d_{i_1,p_1})} < X^{i_2}_{a,q_2(\hat{p},d_{i_2,p_2})} + safety\_h^{i_1,i_2}_{2,\hat{p}}$ )
                                {
                                    $train1_c = i_1$ ; $train2_c = i_2$ ; $type_c = 2$;
                                    $time_c = \min\{X^{i_1}_{d,q_1(\hat{p},d_{i_1,p_1})}, X^{i_2}_{d,q_1(\hat{p},d_{i_2,p_2})}\}$ ;
                                    $t1_c = X^{i_2}_{a,q_2(\hat{p},\hat{d})} - X^{i_1}_{d,q_1(\hat{p},\hat{d})}$ ; $t2_c = X^{i_1}_{a,q_2(\hat{p},\hat{d})} - X^{i_2}_{d,q_1(\hat{p},\hat{d})}$ ;
                                    $resolved_c = 0$
                                    $d\_train1_c = d_{i_1,p_1}$ ;$d\_train2_c = d_{i_2,p_2}$ ; $segment_c = \hat{p}$ ;$c = c+1$
                                }
```

Fig. 4 *Find_Conflicts* module

Basically, the local search heuristic considers shifting a resolved conflict to its neighbors and updates the currently best solution. This process iterates until convergence. We make the following major modifications. When evaluating a neighbor of a resolved conflict that is currently tested for improvement, the proposed heuristic uses the depth-first search branch-and-bound algorithm in Higgins et al. (1996a) to obtain the first feasible train schedule with two possible branching rules (least-lower-bound and least-delay-time). The branch-and-bound-based local search heuristic with the least-lower-bound branching rule is denoted as BBLSH-LLB; the heuristic with the least-delay-time branching rule is denoted as BBLSH-LDT. The proposed heuristics require that the resolved conflicts, which occur after the currently considered conflict, are canceled. In contrast, the original local search heuristic (LSH) maintains all other resolved conflicts unchanged and resolves any new conflict to the nearest siding when it evaluates a neighbor of the currently considered conflict. Furthermore, we provide the detailed implementation of the branch-and-bound-based local search heuristic algorithms that explicitly shows (i) the neighborhood definitions of overtaking and crossing conflicts, (ii) the procedure to check whether any two trains traversing a common segment are in an overtaking or crossing conflict, and (iii) a branch-and-bound-based recursive procedure to generate a feasible train schedule.

The additional notations are provided, followed by the basic modules. The generation of an initial resolved train schedule and the branch-and-bound-based local search heuristic are presented.

4.1 Additional Notations

4.1.1 Sets

C set of (unresolved or resolved) conflicts

4.1.2 Parameters

num_line_p 1 for single-track line; 2 for double-track line

$h^{i,j}_{1,p}$ safety headway between trains $i,j \in I$ traversing in the same direction on segment $p \in P$

$h^{i,j}_{2,p}$ safety headway between trains $i,j \in I$ traversing in opposite directions on segment $p \in P_1$

4.1.3 Decision Variables

ES^i_q extra stop time for train $i \in I$ at station $q \in Q(i)$ (The extra stop times are used to delay trains at stations/sidings to resolve conflicts.)

v^i_p average velocity of train $i \in I$ on segment $p \in P(i)$

$resolved_c$ 1 if conflict $c \in C$ is resolved; 0 otherwise

$type_c$ conflict type (1 for overtaking and 2 for crossing)

$time_c$ conflict time (For unresolved conflicts, $time_{\mathrm{c}}$= minimum of the departure times of two trains from the origin sidings of the conflict segment; for resolved conflicts, $time_{\mathrm{c}}$= arrival time of the delayed train at the conflict siding plus the scheduled stop time)

$train1_c, train2_c$ two trains involved in conflict $c \in C$

$t1_c, t2_c$ delay times associated with $train1_c$, $train2_c$, respectively, to resolve conflict c

$segment_c$ segment where conflict c occurs (the two trains intersect)

d_train1_c, d_train2_c directions of the two trains at $c \in C$ on the conflict segment

$siding_c$ siding where the delayed train is delayed because of $c \in C$

$num\_resolved\_c$ number of resolved conflicts

ROW_train_c train with right-of-way involved in $c \in C$

$Delayed\_train_c$ delayed train involved in $c \in C$

$d\_ROW\_train_c$ direction of the right-of-way train involved in $c \in C$ on the conflict segment

$d\_Delayed\_train_c$ direction of the delayed train involved in $c \in C$ on the conflict segment

z^{LB} lower bound of the optimal objective value

z^{UB} upper bound of the optimal objective value

z^{best} objective value of the currently best solution

4.2 Basic Modules

Three basic modules are described: *Determine_Schedule*, *Find_Conflicts*, and *Find_and_Resolve_Conflicts*.

```
Find_and_Resolve_Conflicts({ $X^i_{a,q}$ },{ $X^i_{d,q}$ }, num_resolved_c)
Call Find_Conflicts({ $X^i_{a,q}$ },{ $X^i_{d,q}$ }, num_resolved_c)
If there is an unresolved conflict; i.e. |C|> num_resolved_c, then
  Consider the unresolved conflict c with the earliest conflict time.
  If the priority weights of $train1_c$ and $train2_c$ are the same, then
      If branching_rule = 0 (Choose the lowest-objective-value branch)
        Two cases are considered:
        case (i) $train1_c$ is delayed. $ES^{train1_c}_{q_1(segment_c, d\_train1_c)} += t1_c + safety\_h^{train1_c, train2_c}_{type_c, segment_c}$ .
              $siding_c = q_1(segment_c, d\_train1_c)$ . Call Determine_Schedule to obtain $z_{case1}$
        case (ii) $train2_c$ is delayed. $ES^{train2_c}_{q_1(segment_c, d\_train2_c)} += t2_c + safety\_h^{train1_c, train2_c}_{type_c, segment_c}$ .
              $siding_c = q_1(segment_c, d\_train2_c)$ . Call Determine_Schedule to obtain $z_{case2}$ .
        Choose the case with the lowest objective value: set $z^{LB} = \min\{ z_{case1}, z_{case2} \}$.
      Else If branching_rule = 1 (a greedy rule chooses the least-delay-time branch)
        If $t1_c \le t2_c$
            $train1_c$ is delayed. $ES^{train1_c}_{q_1(segment_c, d\_train1_c)} += t1_c + safety\_h^{train1_c, train2_c}_{type_c, segment_c}$ and
            $siding_c = q_1(segment_c, d\_train1_c)$ . Call Determine_Schedule to obtain $z^{LB}$ .
        Else if $t1_c > t2_c$
            $train2_c$ is delayed. $ES^{train2_c}_{q_1(segment_c, d\_train2_c)} += t2_c + safety\_h^{train1_c, train2_c}_{type_c, segment_c}$ and
            $siding_c = q_1(segment_c, d\_train2_c)$ . Call Determine_Schedule to obtain $z^{LB}$ .
  Else if $train1_c$ has higher priority weight than $train2_c$
      $train2_c$ is delayed. $ES^{train2_c}_{q_1(segment_c, d\_train2_c)} += t2_c + safety\_h^{train1_c, train2_c}_{type_c, segment_c}$ and
      $siding_c = q_1(segment_c, d\_train2_c)$ . Call Determine_Schedule to obtain $z^{LB}$ .
  Else if $train2_c$ has higher priority weight than $train1_c$
      $train1_c$ is delayed. $ES^{train1_c}_{q_1(segment_c, d\_train1_c)} += t1_c + safety\_h^{train1_c, train2_c}_{type_c, segment_c}$ and
      $siding_c = q_1(segment_c, d\_train1_c)$ . Call Determine_Schedule to obtain $z^{LB}$ .
  Update { $X^i_{a,q}$ },{ $X^i_{d,q}$ }, $ROW\_train_c$ , $Delayed\_train_c$ , $d\_ROW\_train_c$ ,
  $d\_Delayed\_train_c$ and $siding_c$ accordingly.
  If $z^{LB} < z^{best}$
      Update conflict time of this resolved conflict: $time_c = X^{Delayed\_train_c}_{a, siding_c} + S^{Delayed\_train_c}_{siding_c}$ .
      Set $resolved_c = 1$ and $num\_resolved\_c$ += 1.
      Call Find_and_Resolve_Conflicts({ $X^i_{a,q}$ },{ $X^i_{d,q}$ }, num_resolved_c)
  Else if $z^{LB} > z^{best}$
    Return 0 (this train schedule is worse than the currently best solution, so it is pruned)
Else if all conflicts are resolved; i.e. |C|= num_resolved_c, then Return 1 (successfully construct a train schedule).
```

Fig. 5 *Find_and_Resolve_Conflicts* module

```
Denote by $s$ the indicator for when a move improves the solution.
Denote by $\hat{c}$ the resolved conflict in $C$ currently being considered.
Denote by $\hat{c}_{z^{best}}$ the resolved conflict in $C$ whose neighbor generates the best solution found.
Step 0: Generate an initial resolved train schedule represented by the set of resolved conflicts
            $C = \{C_1, C_2, ..., C_{|C|}\}$.
        Sort all conflicts in $C$ in ascending order of conflict time.
        The best objective found $z^{best}$ is obtained.  Set $\hat{c}$ =0, $s$=0.
Step 1: Set $\hat{c} = \hat{c}$ +1.
        If $\hat{c}$ > |$C$| and $s$=0, then stop.
        Else if $\hat{c}$ > |$C$| and $s$=1, then set $\hat{c} = \hat{c}_{z^{best}}$, $s$=0 and go to Step 2.
Step 2: $z^{UB}$ = +infinity.
          For each neighbor of conflict $C_{\hat{c}}$
                Cancel all resolved conflicts whose conflict times are greater than $time_{\hat{c}}$; i.e.
                $ES_{siding_c}^{Delayed\_train_c} = 0\ \forall c = \hat{c},...,|C|$; $v_p^{Delayed\_train_c} = \overline{v}_p^{Delayed\_train_c}$ for $p = segment_c$ to
                the last segment in $P$($Delayed\_train_c$); $v_p^{ROW\_train_c} = \overline{v}_p^{ROW\_train_c}$ for $p = segment_c$ to
                the last segment in $P$($ROW\_train_c$).
                Apply the corresponding move to $C_{\hat{c}}$, set $num\_resolved\_c = \hat{c}$.
                Call the Find_and_Resolve_Conflicts module with the least-lower-bound branching rule for
                BBLSH-LLB and the least-delay-time branching rule for BBLSH-LDT to obtain $C'$ (the set of
          resolved
                conflicts in ascending order of conflict time associated with the neighbor) and objective value $z'$.
                If $z' < z^{UB}$, then set $z^{UB} = z'$, $C^* = C'$ and $\hat{c}_{z^{UB}} = \hat{c}$
          End For
          If $z^{UB} < z^{best}$, then set $z^{best} = z^{UB}$, $\hat{c}_{z^{best}} = \hat{c}_{z^{UB}}$, $C = C^*$ and $s$ = 1.
          Go to Step 1.
```

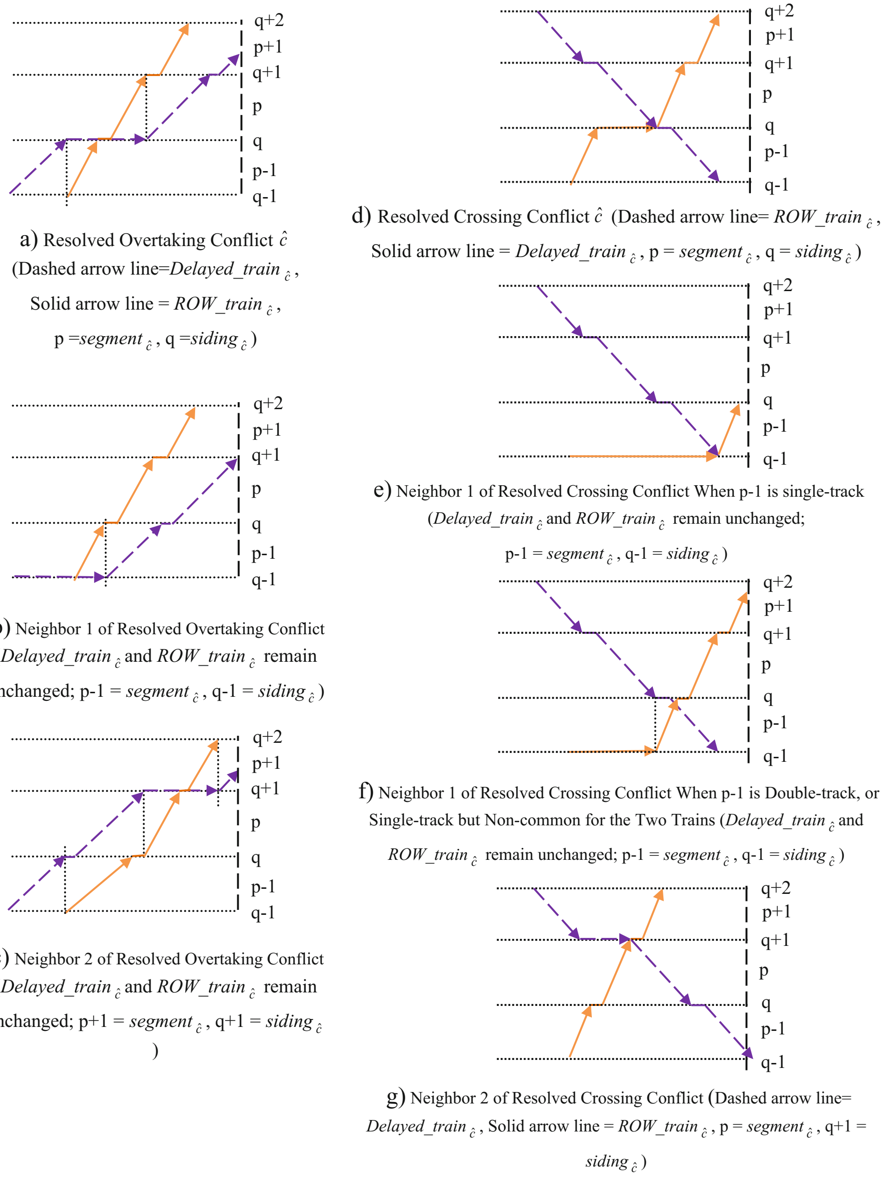
Fig. 6 Pseudo-code of BBLSH-LLB and BBLSH-LDT

4.2.1 *Determine_Schedule Module*

For each train $i \in I$, the departure time from its origin station is X_{dO}^i, train i traverses along rail segments $p \in P(i)$ with velocity v_p^i, and the stop time duration for train i at each siding/station $q \in Q(i)$ is $S_q^i + ES_q^i$. The resulting train schedule is the set of train departure times from and arrival times at sidings/stations, i.e., $\{\{X_{a,q}^i\}, \{X_{d,q}^i\}\}$. This module returns the objective value z, which is an upper bound of the optimal objective value when the resolved schedule is obtained or a lower bound of the optimal objective value for an unresolved schedule. Figure 2 shows the pseudo-code of this module.

4.2.2 *Find_Conflicts Module*

This module determines the two trains ($train1_c$ and $train2_c$) involved in each conflict c, conflict type ($type_c$=1 for overtaking, 2 for crossing), conflict time ($time_c$), conflict segment ($segment_c$), delayed time $t1_c$ ($t2_c$) when $train1_c$ ($train2_c$) is delayed and $train2_c$ ($train1_c$) has the right-of-way as illustrated in Fig. 3, and directions of the two trains on the conflict segment. Specifically, the pseudo-code of this module is given in Fig. 4. This module determines all possible pairs of conflicts. For example, if three trains are involved in a conflict (trains A, B and C) at the same rail segment, this module will determine three conflicts at this rail segment: (i) trains A and B, (ii) A and C and (iii) B and C.



a) Resolved Overtaking Conflict $\hat{c}$ (Dashed arrow line=*Delayed_train* $_{\hat{c}}$, Solid arrow line = *ROW_train* $_{\hat{c}}$, p =*segment* $_{\hat{c}}$, q =*siding* $_{\hat{c}}$)

b) Neighbor 1 of Resolved Overtaking Conflict (*Delayed_train* $_{\hat{c}}$ and *ROW_train* $_{\hat{c}}$ remain unchanged; p-1 = *segment* $_{\hat{c}}$, q-1 = *siding* $_{\hat{c}}$)

c) Neighbor 2 of Resolved Overtaking Conflict (*Delayed_train* $_{\hat{c}}$ and *ROW_train* $_{\hat{c}}$ remain unchanged; p+1 = *segment* $_{\hat{c}}$, q+1 = *siding* $_{\hat{c}}$)

d) Resolved Crossing Conflict $\hat{c}$ (Dashed arrow line= *ROW_train* $_{\hat{c}}$, Solid arrow line = *Delayed_train* $_{\hat{c}}$, p = *segment* $_{\hat{c}}$, q = *siding* $_{\hat{c}}$)

e) Neighbor 1 of Resolved Crossing Conflict When p-1 is single-track (*Delayed_train* $_{\hat{c}}$ and *ROW_train* $_{\hat{c}}$ remain unchanged; p-1 = *segment* $_{\hat{c}}$, q-1 = *siding* $_{\hat{c}}$)

f) Neighbor 1 of Resolved Crossing Conflict When p-1 is Double-track, or Single-track but Non-common for the Two Trains (*Delayed_train* $_{\hat{c}}$ and *ROW_train* $_{\hat{c}}$ remain unchanged; p-1 = *segment* $_{\hat{c}}$, q-1 = *siding* $_{\hat{c}}$)

g) Neighbor 2 of Resolved Crossing Conflict (Dashed arrow line= *Delayed_train* $_{\hat{c}}$, Solid arrow line = *ROW_train* $_{\hat{c}}$, p = *segment* $_{\hat{c}}$, q+1 = *siding* $_{\hat{c}}$)

Fig. 7 Neighborhood definitions for resolved overtaking conflict and resolved crossing conflict

4.2.3 Find_and_Resolve_Conflicts Module

This module resolves one unresolved conflict at a time in ascending order of conflict time with two possible branching rules until it obtains a resolved train schedule. In each iteration of the recursion, after calling the *Find_Conflicts* module, the earliest conflict is selected with the two trains $i_1 \in I; i_2 \in I$ and the segment $p \in P$ where the conflict occurs. If the two trains have different priorities, the train with the higher priority always has the right-of-way, and the train with the lower priority is delayed. The module solves the relaxed program composed of (1.1), (1.6)–(1.8), and the appropriate subset of fixed binary variables from (1.2)–(1.5). The optimal objective value is a lower bound, and the current unresolved train plan is updated.

If the two trains have the same priority, two branching rules are considered: least-lower-bound (Higgins et al. 1996a) and least-delay-time (Higgins et al. 1997). In the least-lower-bound branching rule, two alternatives are considered: (i) train $i_1 \in I$ is delayed, and train $i_2 \in I$ has the right-of-way, and (ii) train $i_2 \in I$ is delayed, and train $i_1 \in I$ has the right-of-way. For each alternative, the module solves the relaxed program composed of (1.1), (1.6)–(1.8), and the appropriate subset of fixed binary variables from (1.2)–(1.5). The optimal objective values are lower bounds: z_{case1} and z_{case2}. The alternative with the least lower bound is selected as the current unresolved train plan.

In the least-delay-time branching rule, the alternative with the least delay time ($\min\{t1_c, t2_c\}$) is selected, and the train associated with $\min\{t1_c, t2_c\}$ is delayed. The module solves the relaxed program composed of (1.1), (1.6)–(1.8), and the appropriate subset of fixed binary variables from (1.2)–(1.5). The optimal objective value is a lower bound, and the current unresolved train plan is updated.

To solve the relaxed program, which is composed of (1.1), (1.6)–(1.8), and the appropriate subset of fixed binary variables from (1.2)–(1.5), the following procedure can be used instead of directly using a mathematical solver. The delayed train and right-of-way train in a resolving conflict c are determined. The extra stop time of the delayed train at the conflict siding is increased by the corresponding delay time ($t1_c$ or $t2_c$) plus an appropriate safety headway. Then, the *Determine_Schedule* module is performed to update the train schedule and determine the lower bound (z^{LB}) of the optimal objective.

During the recursion, the *Find_and_Resolve_Conflicts* module can terminate when it obtains a resolved train schedule or when it finds that the lower bound z^{LB} is greater than or equal to the best obtained objective z^{best}. In the latter case, the module terminates before obtaining a feasible train schedule because the current unresolved train schedule cannot yield a better solution than the currently best schedule. Figure 5 shows the pseudo-code of this module. It is noted that the priority weight plays a role in the *Find_and_Resolve_Conflicts* module and the *Determine_Schedule* module. As long as the higher priority train is assigned a larger priority weight, the train with the larger priority weight always has the right-of-way in resolving a conflict. Different priority weights affect the objective function calculated in the *Determine_Schedule* module. In this paper, the lowest priority is assigned a priority weight of one, and the priority weight increment is one.

4.3 Initial Resolved Train Scheduling Procedure

The procedure is based on the depth-first search branch-and-bound algorithm in Higgins et al. (1996a). We provide the detailed implementation.

Step 1: Solve the relaxed program composed of (1.1) and (1.6)–(1.8) to obtain an unresolved train plan. Alternatively, the unresolved train plan can be obtained from the *Determine_Schedule* module by setting all extra stop times to zero (i.e., $ES_q^i = 0\ \forall i \in I,\ q \in Q(i)$), all velocities to their maximum achievable average velocities (i.e., $v_p^i = \bar{v}_p^i \quad \forall i \in I, \quad p \in P(i)$), and the departure times from their origin equal to their earliest departure times (i.e., $X_{dO}^i = Y_{dO}^i\ \forall i \in I$).

Table 2 Data for toy problem

a.Rail Segments

Segment	Direction 0 (Begin Station, End Station)	Direction 1 (Begin Station, End Station)	Length (km)
0	(0,1)	(1,0)	45
1	(1,2)	(2,1)	38
2	(2,3)	(3,2)	56
3	(3,4)	(4,3)	86
4	(4,5)	(5,4)	72

b.Train Data

Train	Earliest Departure Time (Hours)	Origin Station	Destination Station	Priority Weight
0	6:40	0	4	1
1	7:40	1	5	1
2	6:10	5	1	1

c.Train-Railway Segment Data

Train	Segment	Max Average Velocity (km/h)	Direction
0	0	60	0
0	1	60	0
0	2	60	0
0	3	60	0
1	1	65	0
1	2	65	0
1	3	50	0
1	4	55	0
2	4	60	1
2	3	60	1
2	2	60	1
2	1	60	1

d.Train-Station Data

Train	Station	Scheduled Stop Time (min.)
0	0	0
0	1	10
0	2	10
0	3	10
0	4	0
1	1	0
1	2	10
1	3	10
1	4	10
1	5	0
2	5	0
2	4	10
2	3	10

Table 2 (continued)

2	2	10
2	1	0

All segments are single-track

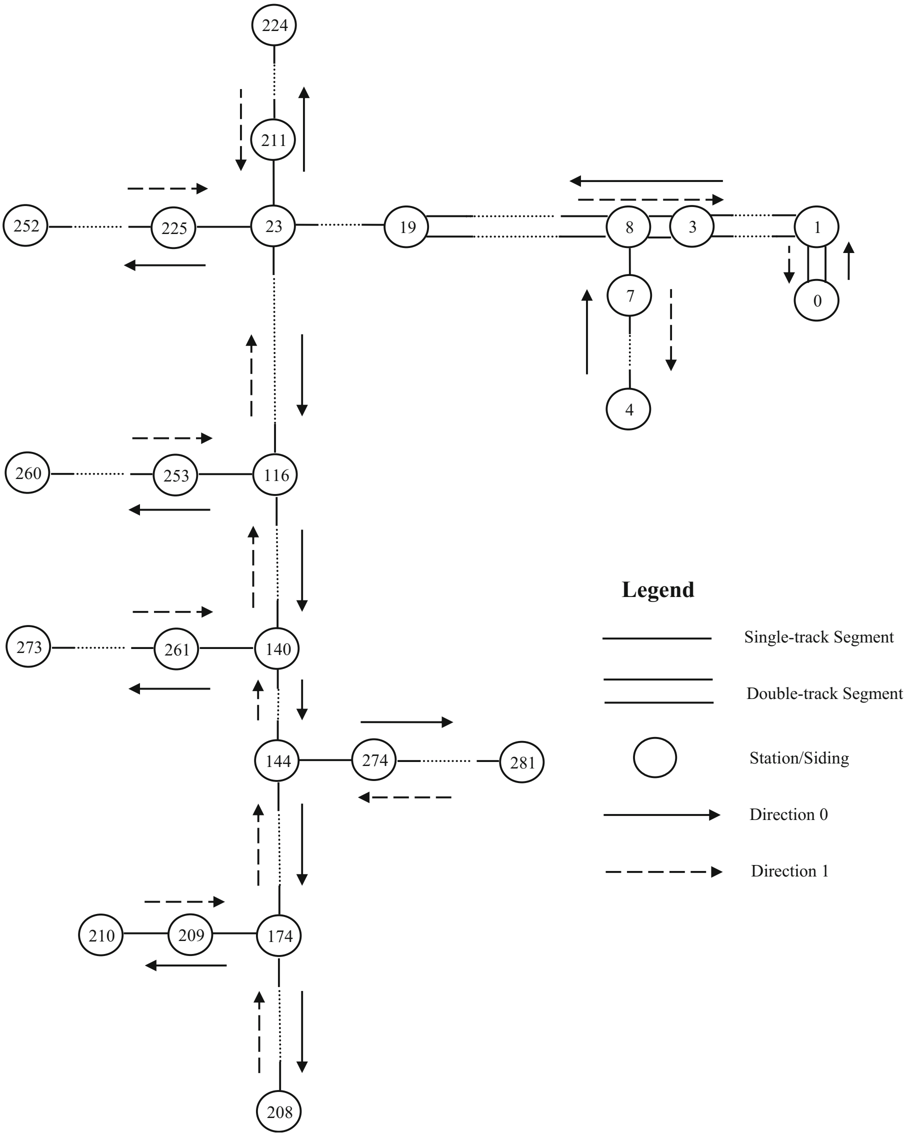


Fig. 8 Thailand southern line railway network

Table 3 Rail segment data for Thailand southern line railway network

Segment	Begin siding	End siding	Length (km)
0	0	1	7
1	1	2	6
2	2	3	5
3	3	8	4
4	4	5	1
5	5	6	1
6	6	7	2
7	7	8	2
8	8	9	3
9	9	10	2
10	10	11	3
11	11	12	5
12	12	13	4
13	13	14	4
14	14	15	4
15	15	16	4
16	16	17	5
17	17	18	4
18	18	19	4
19	19	20	2
20	20	21	5
21	21	22	4
22	22	23	5
23	23	24	4
24	24	25	6
25	25	26	3
26	26	27	5
27	27	28	7
28	28	29	6
29	29	30	5
30	30	31	1
31	31	32	4
32	32	33	6
33	33	34	4
34	34	35	4
35	35	36	4
36	36	37	4
37	37	38	7
38	38	39	5
39	39	40	5
40	40	41	6
41	41	42	10
42	42	43	4
43	43	44	6
44	44	45	5
45	45	46	12
46	46	47	11

Table 3 (continued)

Segment	Begin siding	End siding	Length (km)
47	47	48	4
48	48	49	11
49	49	50	4
50	50	51	4
51	51	52	4
52	52	53	8
53	53	54	3
54	54	55	6
55	55	56	5
56	56	57	8
57	57	58	6
58	58	59	10
59	59	60	8
60	60	61	10
61	61	62	6
62	62	63	7
63	63	64	8
64	64	65	3
65	65	66	5
66	66	67	11
67	67	68	10
68	68	69	3
69	69	70	5
70	70	71	6
71	71	72	8
72	72	73	5
73	73	74	5
74	74	75	6
75	75	76	7
76	76	77	2
77	77	78	7
78	78	79	7
79	79	80	4
80	80	81	6
81	81	82	11
82	82	83	7
83	83	84	6
84	84	85	5
85	85	86	8
86	86	87	7
87	87	88	4
88	88	89	5
89	89	90	6
90	90	91	4
91	91	92	8
92	92	93	9
93	93	94	6
94	94	95	5

Table 3 (continued)

Segment	Begin siding	End siding	Length (km)
95	95	96	8
96	96	97	3
97	97	98	5
98	98	99	5
99	99	100	4
100	100	101	7
101	101	102	8
102	102	103	6
103	103	104	7
104	104	105	6
105	105	106	6
106	106	107	4
107	107	108	8
108	108	109	6
109	109	110	4
110	110	111	10
111	111	112	13
112	112	113	3
113	113	114	5
114	114	115	5
115	115	116	7
116	116	117	4
117	117	118	7
118	118	119	5
119	119	120	5
120	120	121	6
121	121	122	4
122	122	123	8
123	123	124	4
124	124	125	6
125	125	126	4
126	126	127	4
127	127	128	5
128	128	129	7
129	129	130	5
130	130	131	5
131	131	132	7
132	132	133	5
133	133	134	6
134	134	135	7
135	135	136	4
136	136	137	4
137	137	138	4
138	138	139	4
139	139	140	6
140	140	141	5
141	141	142	6
142	142	143	8

Table 3 (continued)

Segment	Begin siding	End siding	Length (km)
143	143	144	5
144	144	145	8
145	145	146	6
146	146	147	11
147	147	148	5
148	148	149	2
149	149	150	3
150	150	151	3
151	151	152	5
152	152	153	4
153	153	154	5
154	154	155	4
155	155	156	3
156	156	157	6
157	157	158	3
158	158	159	4
159	159	160	3
160	160	161	3
161	161	162	6
162	162	163	5
163	163	164	7
164	164	165	4
165	165	166	4
166	166	167	4
167	167	168	5
168	168	169	2
169	169	170	7
170	170	171	7
171	171	172	7
172	172	173	5
173	173	174	7
174	174	175	11
175	175	176	14
176	176	177	11
177	177	178	7
178	178	179	9
179	179	180	11
180	180	181	8
181	181	182	3
182	182	183	6
183	183	184	8
184	184	185	3
185	185	186	1
186	186	187	5
187	187	188	6
188	188	189	3
189	189	190	4
190	190	191	10

Table 3 (continued)

Segment	Begin siding	End siding	Length (km)
191	191	192	3
192	192	193	5
193	193	194	5
194	194	195	9
195	195	196	5
196	196	197	6
197	197	198	7
198	198	199	5
199	199	200	6
200	200	201	5
201	201	202	6
202	202	203	5
203	203	204	4
204	204	205	6
205	205	206	4
206	206	207	7
207	207	208	6
208	174	209	24
209	209	210	21
210	23	211	8
211	211	212	9
212	212	213	4
213	213	214	3
214	214	215	5
215	215	216	3
216	216	217	9
217	217	218	8
218	218	219	6
219	219	220	3
220	220	221	4
221	221	222	5
222	222	223	4
223	223	224	7
224	23	225	4
225	225	226	6
226	226	227	3
227	227	228	13
228	228	229	7
229	229	230	6
230	230	231	11
231	231	232	3
232	232	233	3
233	233	234	2
234	234	235	8
235	235	236	3
236	236	237	3
237	237	238	4
238	238	239	5

Table 3 (continued)

Segment	Begin siding	End siding	Length (km)
239	239	240	3
240	240	241	4
241	241	242	5
242	242	243	5
243	243	244	5
244	244	245	5
245	245	246	2
246	246	247	4
247	247	248	6
248	248	249	1
249	249	250	4
250	250	251	2
251	251	252	3
252	116	253	3
253	253	254	7
254	254	255	4
255	255	256	4
256	256	257	4
257	257	258	3
258	258	259	2
259	259	260	4
260	140	261	9
261	261	262	7
262	262	263	3
263	263	264	5
264	264	265	8
265	265	266	6
266	266	267	6
267	267	268	14
268	268	269	14
269	269	270	4
270	270	271	3
271	271	272	7
272	272	273	7
273	144	274	5
274	274	275	5
275	275	276	4
276	276	277	4
277	277	278	4
278	278	279	3
279	279	280	5
280	280	281	5

Segments 0–3 and 8–18 are double-track; the others are single-track

Step 2: Call the *Find_and_Resolve_Conflicts* Module.

Step 3: The resulted resolved train plan is an initial solution of the branch-and-bound-based local search heuristics. Set the best obtained objective (z^{best}) to the objective value of the initial resolved train schedule.

4.4 Branch-and-Bound-Based Local Search Heuristic Pseudo-Code (BBLSH-LLB and BBLSH-LDT)

BBLSH-LLB and BBLSH-LDT share the same algorithmic steps as follows. A resolved train schedule is represented by the set of resolved conflicts $C = \{C_1, C_2, \ldots, C_{|C|}\}$ in ascending order of conflict time, where $C_c \in C$ is the c^{th} resolved conflict. At the c^{th} resolved conflict, $C_c = \{Delayed\_train_c, ROW\_train_c, d\_Delayed\_train_c, d\_ROW\_train_c, segment_c, siding_c, type_c, time_c\}$. The value $|C|$ represents the total number of resolved conflicts in the schedule. BBLSH-LLB and BBLSH-LDT consider changing a resolved conflict to its neighbors, and the other conflicts that occur after the considered resolved conflict are canceled. Then, the *Find_and_Resolve_Conflicts* module is used to resolve the remaining conflicts. If a neighbor of the considered resolved conflict yields a lower objective value than the best obtained objective (z^{best}), the currently best solution is updated and the ordinal number of the considered resolved conflict is stored in $\hat{c}_{z^{best}}$. In the successive iteration, the proposed local search heuristic starts examining the $\hat{c}_{z^{best}}$ resolved conflict, which is the resolved conflict in C whose neighbor generates the best solution found in the previous iteration. Unlike the original local search heuristic that examines every resolved conflict, the proposed local search heuristic cannot find a better solution in examining the first resolved conflict to the $\hat{c}_{z^{best}}$ resolved conflict (these are already examined in the previous iteration). This is because the original local search heuristic does not cancel any resolved conflicts in the examination of a resolved conflict, whereas the proposed local search heuristic cancels all resolved conflicts that take place after the examined resolved conflict. Specifically, the pseudo-code of BBLSH-LLB and BBLSH-LDT is given in Fig. 6.

Two neighbors are associated with a resolved conflict. For an overtaking conflict (Fig. 7a–c), the first neighbor must shift the delayed train to be delayed at the immediately preceding siding on the route of the delayed train. The second neighbor must shift the delayed train to be delayed at the immediately successive siding on the route of the delayed train. For example, in the resolved overtaking conflict in Fig. 7a, the delayed train is delayed at siding q. The first neighbor of this resolved overtaking conflict must shift the delayed train to be delayed at siding q-1 as shown in Fig. 7b. The second neighbor must shift the delayed train to be delayed at siding $q+1$ as shown in Fig. 7c. Specifically, the two neighbors of a resolved overtaking conflict can be achieved using the two following moves.

- *Move 1 for a Resolved Overtaking Conflict $\hat{c}$* (Fig. 7b)

  Update '$X_{a,q}^{ROW\_train_{\hat{c}}}, X_{d,q}^{ROW\_train_{\hat{c}}}, X_{a,q}^{Delayed\_train_{\hat{c}}}, X_{d,q}^{Delayed\_train_{\hat{c}}}$.

$$ES^{Delayed\_train_{\hat{c}}}_{siding_{\hat{c}}-1} += X^{ROW\_train_{\hat{c}}}_{a,siding_{\hat{c}}} - X^{Delayed\_train_{\hat{c}}}_{d,siding_{\hat{c}}-1} + safety\_h^{Delayed\_train_{\hat{c}},ROW\_train_{\hat{c}}}_{1,segment_{\hat{c}}-1}$$

(Note that $x+=y$ means assigning $x+y$ to the value of x.)

Set $siding_{\hat{c}} = siding_{\hat{c}}-1$, $time_{\hat{c}} = X^{Delayed\_train_{\hat{c}}}_{a,siding_{\hat{c}}} + S^{Delayed\_train_{\hat{c}}}_{siding_{\hat{c}}}$, $segment_{\hat{c}} = segment_{\hat{c}}-1$, $d\_ROW\_train_{\hat{c}} = d_{ROW\_train_{\hat{c}},\ segment_{\hat{c}}}$ and $d\_Delayed\_train_{\hat{c}} = d_{Delayed\_train_{\hat{c}},\ segment_{\hat{c}}}$.

- *Move 2 for a Resolved Overtaking Conflict* $\hat{c}$ (Fig. 7c)

  Update $X^{Delayed\_train_{\hat{c}}}_{a,q}, X^{Delayed\_train_{\hat{c}}}_{d,q}$. Determine the average velocity of the right-of-way train on segment $segment_{\hat{c}}-1$ as shown in Fig. 7c, and update $X^{ROW\_train_{\hat{c}}}_{a,q}, X^{ROW\_train_{\hat{c}}}_{d,q}$.

  $$ES^{Delayed\_train_{\hat{c}}}_{siding_{\hat{c}}+1} += X^{ROW\_train_{\hat{c}}}_{a,siding_{\hat{c}}+2} - X^{Delayed\_train_{\hat{c}}}_{d,siding_{\hat{c}}+1} + safety\_h^{Delayed\_train_{\hat{c}},ROW\_train_{\hat{c}}}_{1,segment_{\hat{c}}+1}.$$

  Set $siding_{\hat{c}} = siding_{\hat{c}}+1$ and $segment_{\hat{c}} = segment_{\hat{c}}+1$. Update $time_{\hat{c}}$, $d\_ROW\_train_{\hat{c}}$, $d\_Delayed\_train_{\hat{c}}$ accordingly.

For a crossing conflict (Fig. 7d–g), the first neighbor must shift the delayed train to be delayed at the immediately preceding siding on the route of the delayed train. The second neighbor must delay the currently right-of-way train at the immediately preceding siding of the conflict segment on the route of the currently right-of-way train, whereas the currently delayed train becomes the right-of-way train. For example, in the resolved crossing conflict in Fig. 7d, the delayed train is delayed at siding q. The first neighbor of this resolved crossing conflict must shift the delayed train to be delayed at siding q-1 as shown in Fig. 7e and f. The second neighbor must delay the right-of-way train in Fig. 7d at siding $q+1$, and the delayed train in Fig. 7d becomes the right-of-way train at the second neighbor as shown in Fig. 7g. Specifically, the two neighbors for a resolved crossing conflict can be achieved using the two following moves.

- *Move 1 for Resolved Crossing Conflict* $\hat{c}$ (Fig. 7e and f)

  If segment p-1 is single-track (Fig. 7e),

  $$ES^{Delayed\_train_{\hat{c}}}_{siding_{\hat{c}}-1} += X^{ROW\_train_{\hat{c}}}_{a,siding_{\hat{c}}-1} - X^{Delayed\_train_{\hat{c}}}_{d,siding_{\hat{c}}-1} + safety\_h^{Delayed\_train_{\hat{c}},ROW\_train_{\hat{c}}}_{2,segment_{\hat{c}}-1}.$$

  If segment p-1 is double-track or single-track but non-common for the two trains (Fig. 7f), $ES^{Delayed\_train_{\hat{c}}}_{siding_{\hat{c}}-1} += X^{ROW\_train_{\hat{c}}}_{a,siding_{\hat{c}}} - X^{Delayed\_train_{\hat{c}}}_{d,siding_{\hat{c}}-1}$.

  Set $segment_{\hat{c}} = segment_{\hat{c}}-1$ and $siding_{\hat{c}} = siding_{\hat{c}}-1$. Update $time_{\hat{c}}$, $d\_ROW\_train_{\hat{c}}$, $d\_Delayed\_train_{\hat{c}}$ accordingly.

- *Move 2 for Resolved Crossing Conflict* $\hat{c}$ (Fig. 7g)
  Swap the delayed train and the ROW train for conflict $\hat{c}$.
  Update $X_{a,q}^{ROW\_train_{\hat{c}}}, X_{d,q}^{ROW\_train_{\hat{c}}}, X_{a,q}^{Delayed\_train_{\hat{c}}}, X_{d,q}^{Delayed\_train_{\hat{c}}}$.

$$ES_{siding_{\hat{c}}+1}^{Delayed\_train_{\hat{c}}} += X_{a,siding_{\hat{c}}+1}^{ROW\_train_{\hat{c}}} - X_{d,siding_{\hat{c}}+1}^{Delayed\_train_{\hat{c}}} + safety\_h_{2,segment_{\hat{c}}}^{Delayed\_train_{\hat{c}},ROW\_train_{\hat{c}}}$$

Set $siding_{\hat{c}} = siding_{\hat{c}} + 1$. Update $time_{\hat{c}}$ accordingly ($segment_{\hat{c}}$, $d\_ROW\_train_{\hat{c}}$ and $d\_Delayed\_train_{\hat{c}}$ remain unchanged).

5 Computational Results

BBLSH-LLB, BBLSH-LDT, LSH and the equivalent manual solution method are implemented in Eclipse Java EE IDE. The manual solution method (without optimization), which is currently adopted at the State Railway of Thailand and is equivalent to the generation of an initial resolved train schedule with the least-delay-time branching rule. The proposed mathematical program is coded in IBM ILOG CPLEX Optimization Studio IDE Version 12.6, which solves the problem with the branch-and-bound algorithm. The CPLEX commercial solver is employed to solve the modified formulation with the relative MIP gap tolerance of 0.000001. This tolerance sets a relative tolerance on the gap between the best integer objective and the objective of the best node remaining (i.e., best lower bound). These run on a computer with 2.50 GHz Intel Core i7-3537U processor and 8 GB of RAM under Windows 8.1. Two test problems are employed: toy problem and Thailand Southern line railway network. The data are first described; then, the computational results are discussed.

5.1 Data

Two test networks are considered: toy network and Thailand Southern line railway network.

5.1.1 Toy Network Data

A toy problem is a railway line composed of three trains, five rail segments and six stations. Table 2 shows the data for the toy problem.

5.1.2 Thailand Southern Line Railway Network Data

Thailand has 4431 km of meter gauge railway tracks (excluding mass transit lines) managed by the State Railway of Thailand. There are four main lines: Northern Line, Northeastern Line, Eastern Line, and Southern Line. Most railway tracks are single-track; in the Southern line railway network (our case study), the double-track distance is 64 km, which is 4.07 % of the total distance (1577 km). The test network is the Southern Line Railway Network in Thailand, as shown in Fig. 8. The network is composed of 266 single-track segments, 15 double-track segments, and 282 stations/

sidings, and the total distance is 1577 km. Table 3 shows the rail segment data. Table 4 shows the train data. There are four train classes. The priority weight of four denotes the highest priority, and the priority weight of one denotes the lowest priority. The maximum achievable average velocity of the trains ($\overline{v}_p^i$) is determined from the following equation, which is used by experienced staffs at the State Railway of Thailand: $\overline{v}_p^i = \frac{l_p}{l_p/v_p^{i,\max} + acceleration\_time_p^i + deceleration\_time_p^i}$, where $v_p^{i,\max}$ is the maximum velocity of the train, and $acceleration\_time_p^i$ and $deceleration\_time_p^i$ are the acceleration and deceleration times for train i on segment p, respectively. The scheduled stop times at stations/sidings vary from zero to 2 min (because of the space limit, these values are not shown). The minimum achievable average velocities ($\underline{v}_p^i$) are assumed to be zero. We assume that $h_{1,p}^{i,j} = h_{2,p}^{i,j} = 3$ sec onds. Four test problems are considered with different numbers of trains: 26 trains (the first 26 trains in Table 4), 50 trains (the first 50 trains in Table 4), 76 trains (the first 76 trains in Table 4) and 108 trains. There are four train classes.

5.2 Experimental Results

The exact solution method (CPLEX solver), the equivalent manual solution method, LSH, BBLSH-LLB and BBLSH-LDT are performed in the toy, 26-train, 50-train, 76-train and 108-train problems. Table 5 shows the best objective values, improvement percentage (compared with the equivalent manual solution), optimality gap (compared with the best lower bound from the CPLEX solver), total travel time, total travel time saving (compared with the equivalent manual solution), total CPU time and number of heuristic iterations on the five test problems. Figure 9 illustrates the best resolved train schedule for the 76-train problem from BBLSH-LLB. Note that the improvement percentage, the optimality gap and the total travel time saving are calculated using the equation:

$$\%\text{improvement} = \frac{100 \times (\text{best obj from heuristic} - \text{best obj from equivalent manual method})}{\text{best obj from equivalent manual method}}$$
$$\%\ \text{optimality}\ \ \text{gap} = \frac{100 \times (\text{best obj from heuristic} - \text{best lower bound from CPLEX})}{\text{best lower bound from CPLEX}}$$
$$\begin{aligned}\text{Total Travel Time Saving} = {} & \text{Total Travel Time of the Equivalent Manual Solution} \\ & - \text{Total Travel Time of Heuristic}\end{aligned}$$

In terms of computational time, the equivalent manual solution method takes the shortest total computational time for all test problems. The exact solution method takes the longest CPU time (4.72×10^{-5}, 3.409, 5.278, 21.425 and 25.339 h) to terminate at the optimality gaps of 0, 2.283, 3.150, 4.594 and 8.387 % for the five problems, respectively. LSH takes the same order of magnitude of total CPU time as the equivalent manual method. BBLSH-LLB and BBLSH-LDT require much longer total CPU times to terminate. The largest computational burden for BBLSH-LLB and BBLSH-LDT is from the recursive sub-procedure *Find_and_Resolve_Conflicts*. LSH takes fewer heuristic iterations to terminate (1, 1, 2, 5 and 6 iterations for the five test problems) than BBLSH-LLB (2, 6, 6, 4 and 7 iterations) and BBLSH-LDT (2, 6, 6, 5 and 7 iterations). Thus, the branch-and-bound-based local search heuristics find better solution with the sacrifice of longer total CPU time when compared with the LSH and the equivalent manual solution method.

Table 4 Train data for Thailand southern line railway network

Train	Earliest departure time (Hours)	Origin station	Destination station	Max velocity (km/h)	Priority weight
0	14:45	0	210	90	4
1	17:40	210	0	90	4
2	15:10	0	208	90	4
3	14:20	208	0	90	4
4	22:50	0	117	120	4
5	10:40	117	0	120	4
6	22:50	0	190	120	4
7	14:55	190	0	120	4
8	8:05	0	117	120	4
9	20:25	117	0	120	4
10	17:05	0	269	90	4
11	17:20	269	0	90	4
12	19:30	0	281	90	4
13	15:00	281	0	90	4
14	18:40	0	273	90	3
15	12:40	273	0	90	3
16	15:35	0	190	90	3
17	12:35	190	0	90	3
18	13:00	0	208	90	3
19	11:30	208	0	90	3
20	17:35	0	281	90	3
21	13:00	281	0	90	3
22	6:30	174	208	90	3
23	14:55	208	174	90	3
24	19:30	4	101	90	3
25	19:55	101	4	90	3
26	13:05	4	63	90	2
27	4:45	63	4	90	2
28	5:25	101	4	90	2
29	7:25	4	101	90	2
30	7:45	4	252	90	2
31	12:55	252	4	90	2
32	13:55	4	252	90	2
33	5:20	252	4	90	2
34	9:20	0	49	105	2
35	14:10	49	0	105	2
36	17:50	4	31	90	4
37	5:00	31	4	90	4
38	16:40	0	224	105	4
39	4:40	224	0	105	4
40	6:35	90	174	90	2
41	6:30	174	90	90	2
42	6:20	117	208	90	2
43	6:30	208	117	90	2
44	6:00	281	208	90	2
45	8:55	208	281	90	2
46	6:30	190	208	90	2

Table 4 (continued)

Train	Earliest departure time (Hours)	Origin station	Destination station	Max velocity (km/h)	Priority weight
47	15:20	208	190	90	2
48	9:38	281	190	90	2
49	6:30	190	281	90	2
50	14:20	281	157	90	2
51	6:00	157	281	90	2
52	6:00	157	208	90	2
53	12:20	208	157	90	2
54	4:25	23	252	70	2
55	15:15	252	23	70	2
56	16:55	117	260	70	1
57	6:00	260	117	70	1
58	6:30	0	252	105	3
59	14:25	252	0	105	3
60	6:30	0	49	105	3
61	15:20	49	0	105	3
62	10:50	232	252	90	3
63	12:55	252	232	90	3
64	17:50	0	210	75	4
65	13:45	210	0	75	4
66	14:20	174	210	105	4
67	8:40	210	174	105	4
68	6:45	0	72	90	3
69	9:00	72	0	90	3
70	8:40	232	234	25	3
71	10:25	232	234	25	3
72	11:20	232	234	25	3
73	11:45	233	234	25	3
74	12:00	233	234	25	3
75	12:15	233	234	25	3
76	12:30	233	234	25	3
77	12:45	233	234	25	3
78	13:00	233	234	25	3
79	13:15	233	234	25	3
80	14:45	232	234	25	3
81	15:10	233	234	25	3
82	17:43	232	234	25	3
83	8:56	234	232	25	3
84	10:42	234	232	25	3
85	11:37	234	233	25	3
86	11:53	234	233	25	3
87	12:08	234	233	25	3
88	12:23	234	233	25	3
89	12:38	234	233	25	3
90	12:53	234	233	25	3
91	13:08	234	233	25	3
92	13:23	234	232	25	3
93	15:02	234	233	25	3

Table 4 (continued)

Train	Earliest departure time (Hours)	Origin station	Destination station	Max velocity (km/h)	Priority weight
94	15:18	234	232	25	3
95	17:59	234	232	25	3
96	20:40	1	174	55	1
97	13:15	174	1	55	1
98	7:30	174	210	55	1
99	9:55	210	174	55	1
100	1:05	1	174	55	2
101	15:10	174	1	70	2
102	0:30	1	174	55	1
103	0:05	174	1	55	1
104	10:00	1	210	60	1
105	15:50	210	1	60	1
106	1:10	1	23	55	3
107	19:45	23	1	55	3

In terms of solution quality, the equivalent manual solution method has the worst performance for all test problems. LSH yields its best solutions with 0, 0, 0.00129, 0.02408 and 0.04298 % improvement (the total travel time savings were 0, 0, 1.166, 32.963 and 62.717 min) and 3.165, 3.690, 4.213, 5.823 and 8.796 % optimality gap for the toy, 26-train, 50-train, 76-train and 108-train problems, respectively. BBLSH-LLB and BBLSH-LDT yield the same best solutions with 2.827, 0.7248 and 0.6328 % improvement (the total travel time savings were 28.00, 355.33 and 388.50 min) and 0.2484, 2.939 and 3.555 % optimality gap for the toy, 26-train and 50-train problems, respectively. For the 76-train problem, BBLSH-LLB (with 0.5659 % improvement, 5.249 % optimality gap, total travel time saving = 372.7 min, and total CPU time =2.548 h) outperforms BBLSH-LDT (with 0.5275 % improvement, 5.290 % optimality gap, total travel time saving = 354.67 min and total CPU time = 2.578 h) on both solution quality and total CPU time. Similarly, for the 108-train problem, BBLSH-LLB (with 0.80752 % improvement, 7.963 % optimality gap, total travel time saving = 885.27 min and total CPU time = 12.442 h) outperforms BBLSH-LDT (with 0.61497 % improvement, 8.173 % optimality gap, total travel time saving = 642.42 min and total CPU time = 13.160 h) on both solution quality and total CPU time. Figure 10 shows the convergence characteristics of the exact solution method and the three heuristics for the 26-train, 50-train, 76-train and 108-train problems. For all test problems, LSH converges notably fast to worse solutions than BBLSH-LLB and BBLSH-LDT. BBLSH-LLB converges faster than BBLSH-LDT on all test problems. By the time BBLSH-LLB and BBLSH-LDT converge, the CPLEX solver just finds much worse solutions. Figure 11 shows the relationships of the total CPU time and problem size for LSH, BBLSH-LLB and BBLSH-LDT. From our empirical results, BBLSH-LLB and BBLSH-LDT appear to be of polynomial time because the total CPU time is a polynomial expression in terms of the problem size.

Table 5 Best obtained solutions by exact solution method, equivalent manual solution method, LSH, BBLSH-LLB and BBLSH-LDT for toy problem and Thailand southern line railway network

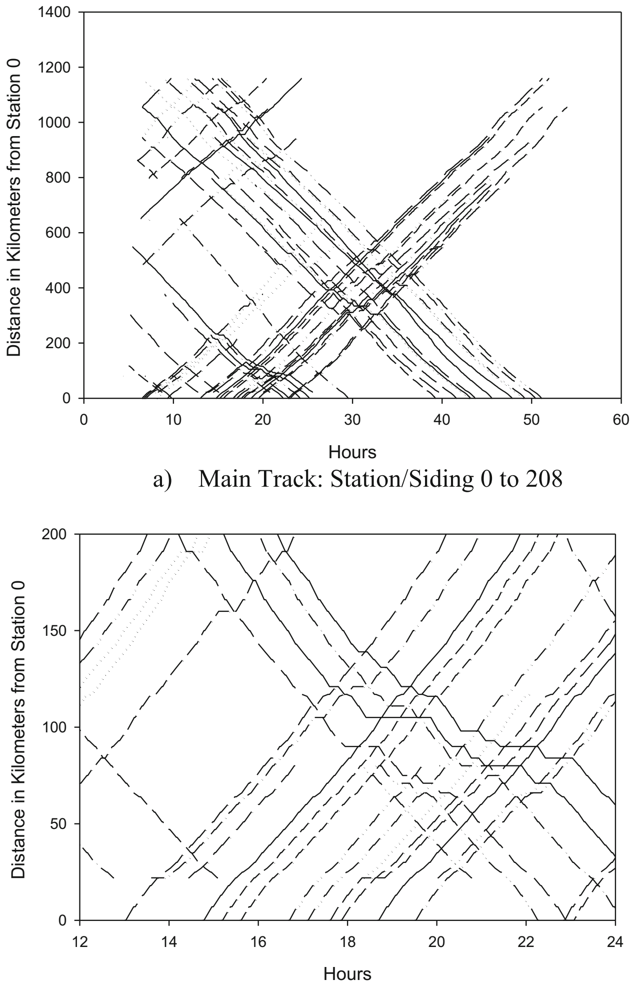
Method	Solution characteristics	Toy problem	Thailand southern line railway network test problem			
			26-train problem	50-train problem	76-train problem	108-train problem
Exact Solution (CPLEX)	Best Objective	960.053	146176.50	178371.93	214301.51	252752.1
	Optimality Gap	0 %	2.283 %	3.150 %	4.594 %	8.387 %
	Total CPU Time (Hours)	4.72×10^{-5}	3.409	5.278	21.425	25.339
	Best Lower Bound	960.053	142913.57	172925.36	204889.45	233194.26
Equivalent Manual Solution Method	Best Objective	990.438	148187.50	180212.69	216872.23	253814.23
	Optimality Gap	3.165 %	3.690 %	4.214 %	5.848 %	8.842 %
	Total Travel Time (Minutes)	990.438	41818.33	56693.69	68502.42	95128.87
	Total CPU Time (Hours)	0.889×10^{-5}	0.00107	0.00311	0.00808	0.02426
LSH	Best Objective (% Improvement)	990.438 (0 %)	148187.50 (0 %)	180210.357 (0.00129 %)	216820.010 (0.02408 %)	253705.147 (0.04298 %)
	Optimality Gap	3.165 %	3.690 %	4.213 %	5.823 %	8.796 %
	Total Travel Time (Minutes)	990.438	41818.33	56692.524	68469.457	95066.153
	Total Travel Time Saving (Minutes)	0	0	1.166	32.963	62.717
	Total CPU Time (Hours)	1.61×10^{-5}	0.00157	0.00602	0.02043	0.10623
	Number of Iterations	1	1	2	5	6
BBLSH-LLB (Least-Lower-Bound Branching Rule)	Best Objective (% Improvement)	962.438 (2.827 %)	147113.50 (0.7248 %)	179072.36 (0.6328 %)	215644.86 (0.5659 %)	251764.635 (0.80752 %)
	Optimality Gap	0.2484 %	2.939 %	3.555 %	5.249 %	7.963 %
	Total Travel Time (Minutes)	962.438	41463.00	56305.19	68129.72	94243.596
	Total Travel Time Saving (Minutes)	28.00	355.33	388.5	372.7	885.274
	Total CPU Time (Hours)	1.78×10^{-5}	0.279	1.004	2.548	12.442
	Number of Iterations	2	6	6	4	7
BBLSH-LDT (Least-Delay-Time Branching Rule)	Best Objective (% Improvement)	962.438 (2.827 %)	147113.50 (0.7248 %)	179072.36 (0.6328 %)	215728.30 (0.5275 %)	252253.350 (0.61497 %)
	Optimality Gap	0.2484 %	2.939 %	3.555 %	5.290 %	8.173 %
	Total Travel Time (Minutes)	962.438	41463.00	56305.19	68147.75	94486.453
	Total Travel Time Saving (Minutes)	28.00	355.33	388.5	354.67	642.417
	Total CPU Time (Hours)	2.67×10^{-5}	0.279	1.022	2.578	13.160
	Number of Iterations	2	6	6	5	7

- % Optimality Gap = 100 × (best objective from the heuristic – best lower bound from the exact solution)/best lower bound from the exact solution
- % Improvement = 100 × (best objective from the heuristic – best objective from the equivalent manual solution method)/best objective from the equivalent manual solution method
- Total travel time saving of an algorithm = total travel time of the algorithm – total travel time of the equivalent manual solution method

The computational time of the branch-and-bound-based local search heuristics in the 108-train problem may be improved with the trade-off of the solution quality. Two sampling strategies for selecting the resolved conflicts to be tested for improvement in BBLSH-LLB (Step 2 in Fig. 6) are considered. First, BBLSH-LLB-ES denotes the BBLSH-LLB algorithm that selects every second resolved conflict in the main iteration of local search heuristic. The sampling strategy in BBLSH-LLB-ES allows the algorithm to examine only half of the resolved conflicts from the uniform distribution. Second, BBLSH-LLB-ET denotes the BBLSH-LLB algorithm that selects every third resolved conflict in the main iteration of local search heuristic. BBLSH-LLB-ET examines one-third of the resolved conflicts from the uniform distribution. Table 6 shows the results of BBLSH-LLB-ES and BBLSH-LLB-ET in the 108-train problem. As expected, the total CPU time (6.504 h) by BBLSH-LLB-ES is decreased by approximately half of that (12.442) by BBLSH-LLB. The solution quality (8.070 % optimality gap) by BBLSH-LLB-ES is slightly worse than that (7.963 % optimality gap) by BBLSH-LLB. Interestingly, BBLSH-LLB-ET yields marginally worse solution than BBLSH-LLB-ES, whereas BBLSH-LLB-ET can save the total CPU time by 1.022 h. It is noted that the abbreviations for different local search heuristic algorithms developed in this article are summarized in Table 7.

6 Summary and Conclusions

We consider the train-scheduling problem in a single-track railway network in the planning application. The formulation for a rail line in Higgins et al. (1996a) was modified to accommodate a railway network. Two proposed branch-and-bound-based local search heuristic algorithms are based on the respective least-lower-bound branching rule (Higgins et al. 1996a) and least-delay-time branching rule (Higgins et al. 1997). The proposed branch-and-bound-based local search heuristic algorithms consider changing a resolved conflict to its neighbors, and the other conflicts that occur after the considered resolved conflict are canceled. Then, the proposed algorithms resolve the remaining conflicts with the two branching rules. In resolving any conflict between trains *i* and *j* with the least-lower-bound branching rule, two cases (case 1: delayed *i* and right-of-way *j*; case 2: delayed *j* and right-of-way *i*) are assessed in terms of the lower bound of objective function. The case with delayed *i* and right-of-way *j* is chosen if the lower bound of objective function associated with delaying train *i* is less than the lower bound associated with delaying train *j*. On the other hand, in resolving a conflict with the least-delay-time branching rule, the two cases are assessed in terms of local delay time. The case with delayed *i* and right-of-way *j* is chosen if the local delay time associated with delaying train *i* is less than that associated with delaying train *j*; i.e., a conflict is resolved to the nearest siding. The detailed implementation of the two proposed heuristic algorithms is presented. The test networks are the toy network and the southern line railway network in Thailand, which is composed of 266 single-track segments, 15 double-track segments, 282 stations/sidings and a total distance of 1577 km. Four test problems on the Thailand southern line railway network are considered with different numbers of multi-class trains: 26-train, 50-train, 76-train and 108-train. The exact solution method (CPLEX solver), the two proposed heuristic



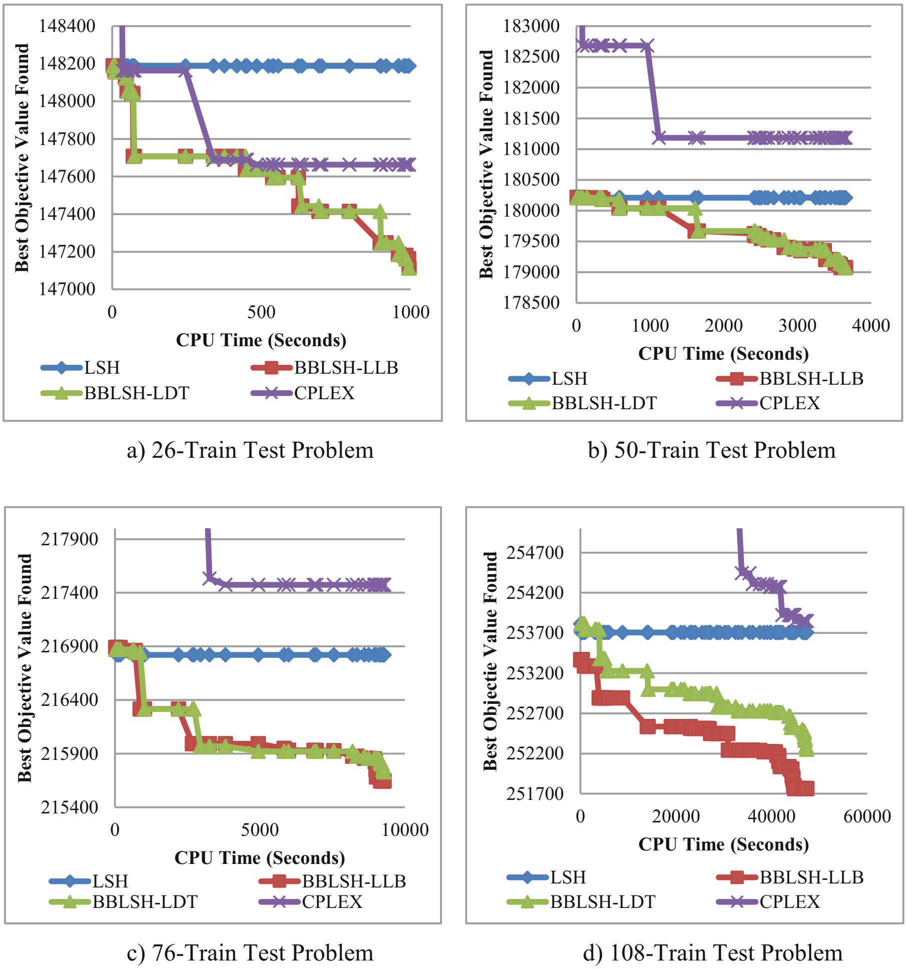
a) Main Track: Station/Siding 0 to 208

b) Main Track: Station/Siding 0-45 and Between 12:00 and 24:00 Hours

Fig. 9 Best resolved train schedule found in 76-train test problem

algorithms, the original local search heuristic, and the equivalent manual solution method used in the State Railway of Thailand are performed for the five test problems. The results are compared in terms of the solution quality, computational time, and convergence speed.

In terms of computational time, the equivalent manual solution method and the original local search heuristic take the same order of magnitude of total CPU time. The proposed heuristic algorithms require longer total CPU time and more heuristic iterations to terminate than the original local search heuristic, which allows the proposed heuristic algoritysms to find better solution with the sacrifice of longer total CPU time. The exact solution method spends the longest time to determine solutions with the known optimality gaps. In terms of solution quality, the original local search heuristic yields its best solutions with up to 0.04298 % improvement and up to 8.796 % optimality gap compared to the equivalent manual solutions for the five problems.



a) 26-Train Test Problem

b) 50-Train Test Problem

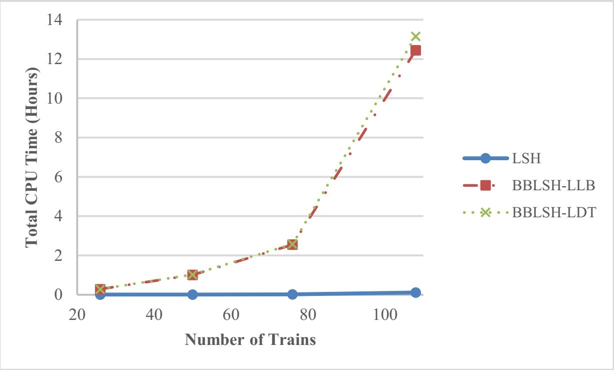
c) 76-Train Test Problem

d) 108-Train Test Problem

Fig. 10 Convergence characteristics of exact solution method (CPLEX Solver), local search heuristic and two branch-and-bound-based local search heuristics in 26-train, 50-train, 76-train and 108-train problems

The proposed heuristic algorithm with the least-lower-bound branching rule yields its best solutions with up to 2.827 % improvement and up to 7.963 % optimality gap. The proposed heuristic algorithm with the least-delay-time branching rule yields its best solutions with up to 2.827 % improvement and up to 8.173 % optimality gap. For all test problems, the proposed heuristic algorithm with the least-lower-bound branching rule converges faster than that with the least-delay-time branching rule. From the empirical results, the proposed heuristic algorithms appear to be solvable in polynomial time.

Two sampling strategies based on the uniform distribution are incorporated into the proposed heuristic algorithm with the least-lower-bound branching rule in order to shorten the computational time. The first sampling strategy allows the algorithm to examine only half of the resolved conflicts from the uniform distribution; i.e., the algorithm examines every second resolved conflict in the local search iteration. The



Heuristic	Regression	Adjusted R^2
BBLSH-LLB	CPU in Hours = 4.638 – 0.227(No. of Trains) +0.002758(No. of Trains)^2	0.9568
BBLSH-LDT	CPU in Hours = 5.045 – 0.247(No. of Trains) +0.002967(No. of Trains)^2	0.9537

Fig. 11 Relationships of total CPU time and problem size for local search heuristic and two branch-and-bound-based local search heuristics

second sampling strategy allows the algorithm to examine one-third of the resolved conflicts from the uniform distribution; i.e., the algorithm examines every third resolved conflict in the local search iteration. The proposed heuristic algorithm with the least-lower-bound branching rule and the two sampling strategies is performed in the 108-train problem. The best solutions from the two sampling strategies are almost the same, and slightly worse than the algorithm without a sampling strategy. The two sampling strategies can save the total CPU time by about 50 % when compared with the algorithm without a sampling strategy.

This paper can be extended in multiple directions. The immediate improvement of the proposed formulation is to consider that trains can be delayed at a feasible siding/

Table 6 Best obtained solutions by BBLSH-LLB-ES and BBLSH-LLB-ET for 108-train problem

Method	Solution characteristics	108-train problem
BBLSH-LLB-ES	Best objective (% Improvement)	252011.92 (0.7101 %)
	Optimality gap	8.070 %
	Total Travel Time (Minutes)	94403.81
	Total Travel Time Saving (Minutes)	725.06
	Total CPU Time (Hours)	6.504
	Number of Iterations	5
BBLSH-LLB-ET	Best objective (% Improvement)	252022.23 (0.7060 %)
	Optimality Gap	8.073 %
	Total Travel Time (Minutes)	94438.04
	Total Travel Time Saving (Minutes)	690.83
	Total CPU Time (Hours)	5.482
	Number of Iterations	4

Table 7 Abbreviations for different local search heuristic algorithms

Abbreviation	Description
LSH	Local search heuristic in Higgins et al. (1997)
BBLSH-LLB	Branch-and-bound-based local search heuristic with the least-lower-bound branching rule. LSH in Higgins et al. (1997) is modified such that when evaluating a neighbor of a resolved conflict that is currently tested for improvement, BBLSH-LLB uses the depth-first search branch-and-bound algorithm in Higgins et al. (1996a) to obtain the first feasible train schedule with the branching rule that selects the neighbor with the least lower bound of objective function.
BBLSH-LDT	Branch-and-bound-based local search heuristic with the least-delay-time branching rule. LSH in Higgins et al. (1997) is modified such that when evaluating a neighbor of a resolved conflict that is currently tested for improvement, BBLSH-LDT uses the depth-first search branch-and-bound algorithm in Higgins et al. (1996a) to obtain the first feasible train schedule with the branching rule that selects the neighbor with the least delay time.
BBLSH-LLB-ES	BBLSH-LLB algorithm that selects every second resolved conflict in the main iteration of local search heuristic; the sampling strategy in BBLSH-LLB-ES allows the algorithm to examine only half of the resolved conflicts from the uniform distribution.
BBLSH-LLB-ET	BBLSH-LLB algorithm that selects every third resolved conflict in the main iteration of local search heuristic; the sampling strategy in BBLSH-LLB-ET allows the algorithm to examine one-third of the resolved conflicts from the uniform distribution.

station within the maximum allowable extra delayed times. In the proposed formulation and heuristics, two parameters may be used: $\delta_q^i = 1$ if train $i \in I$ can be delayed at station $q \in Q(i)$ and 0 otherwise; $UB\_ES_q^i$ = maximum allowable extra stop time for train $i \in I$ at station $q \in Q(i)$. Then, the constraint set (1.8) may be replaced by the following constraints:

$$X_{a,q}^i + S_q^i + ES_q^i = X_{d,q}^i \quad \forall i \in I; q \in Q(i) \tag{1.9}$$

$$0 \le ES_q^i \le \delta_q^i \cdot UB\_ES_q^i \quad \forall i \in I; q \in Q(i) \tag{1.10}$$

Constraint set (1.9) states that the train departure time from a station/siding is equal to the summation of the train arrival time, scheduled stop time and extra stop time at this station. Constraint set (1.10) enforces that the extra stop time is non-negative and may not be greater than the maximum allowable extra stop time. If train i cannot be delayed at station/siding q, then δ_q^i is equal to zero, which yields $ES_q^i = 0$. If train i can be delayed at station/siding q, then δ_q^i is one, which yields $ES_q^i \le UB\_ES_q^i$. The proposed branch-and-bound-based local search heuristics can be modified to accommodate the additional constraints. The branch-and-bound-based local search heuristics may be used as a sub-procedure in a metaheuristic algorithm. The extension can be made for the multi-criteria train timetabling problem.

Acknowledgments The authors gratefully acknowledge the support from the Department of Civil Engineering, King Mongkut's University of Technology Thonburi under contract number CE-KMUTT-5801 and the Thailand Research Fund. The authors would like to thank the staffs at the State Railway of Thailand for the test data and the anonymous reviewers for their valuable comments and suggestions to help us improve the manuscript.

References

Adenso-Diaz B, Gonzalez MO, Gonzalez-Torre P (1999) On-line timetable re-scheduling in regional train services. Transp Res B 33(6):387–398

Albrecht T (2009) The influence of anticipating train driving on the dispatching process in railway conflict situations. Netw Spat Econ 9(1):85–101

Assad A (1980) Models for rail transportation. Transp Res A 14:205–220

Brannlund U, Lindberg PO, Nou A, Nilsson JE (1998) Railway timetabling using Lagrangian relaxation. Transp Sci 32:358–369

Bussieck MR, Kreuzer P, Zimmermann UT (1997) Optimal lines for railway systems. Eur J Oper Res 96(1): 54–63

Cacchiani V, Caprara A, Toth P (2008) A column generation approach to train timetabling on a corridor. 4OR6 (2): 125–142

Cacchiani V, Caprara A, Toth P (2010a) Non-cyclic train timetabling and comparability graphs. Oper Res Lett 38(3):179–184

Cacchiani V, Caprara A, Toth P (2010b) Scheduling extra freight trains on rail networks. Transp Res B 44(2): 215–231

Cai X, Goh CJ, Mees AI (1998) Greedy heuristics for rapid scheduling of trains on a single track. IIE Trans 30(5):481–493

Caimi G, Fuchsberger M, Laumanns M, Luthi M (2012) A model predictive control approach for discrete-time rescheduling in complex central railway station areas. Comput Oper Res 39(11):2578–2593

Caprara A, Fischetti M, Toth P (2002) Modeling and solving the train timetabling problem. Oper Res 50(5): 851–861

Carey M (1994a) A model and strategy for train pathing with choice of lines, platforms and routes. Transp Res B 28(5):333–353

Carey M (1994b) Extending a train pathing model from one-way to two-way track. Transp Res B 28(5):395–400

Carey M, Crawford I (2007) Scheduling trains on a network of busy complex stations. Transp Res B 41(2): 159–178

Carey M, Lockwood D (1995) A model, algorithms and strategy for train pathing. J Oper Res Soc 46(8):988–1005

Castillo E, Gallego I, Urena JM, Coronado JM (2011) Timetabling optimization of a mixed double- and single-track rail network. Appl Math Model 35(2):859–878

Chang YH, Yeh CH, Shen CC (2000) A multiobjective model for passenger train services planning: application to Taiwan's high-speed rail line. Transp Res Part B 34:91–106

Chen B, Harker PT (1990) Two moments estimation of the delay on single-track rail lines with scheduled traffic. Transp Sci 24(4):261–275

Cordeau JF, Toth P, Vigo D (1998) A survey of optimization models for train routing and scheduling. Transp Sci 32:380–404

Corman F, D'Ariano A, Pacciarelli D, Pranzo M (2010) A tabu search algorithm for rerouting trains during rail operations. Transp Res B 44(1):175–192

Corman F, D'Ariano A, Hansen IA, Pacciarelli D (2011) Optimal multi-class rescheduling of railway traffic. J Rail Transp Plan Manage 1(1):14–24

D'Ariano A, Pranzo M (2009) An advanced real-time train dispatching system for minimizing the propagation of delays in a dispatching area under severe disturbances. Netw Spat Econ 9(1):63–84

D'Ariano A, Pacciarelli D, Pranzo M (2007) A branch and bound algorithm for scheduling trains in a railway network. Eur J Oper Res 183:643–657

D'Ariano A, Corman F, Pacciarelli D, Pranzo M (2008) Reordering and local rerouting strategies to manage train traffic in real time. Transp Sci 42(4):405–419

Dorfman M, Medanic J (2004) Scheduling trains on a rail network using a discrete event model of rail traffic. Transp Res B 38(1):81–98

Greenberg HH (1968) A branch and bound solution to the general scheduling problem. Oper Res 16:352–261

Han L, Luong BT, Ukkusuri S (2015) An algorithm for the one commodity pickup and delivery traveling salesman problem with restricted depot. Netw Spat Econ. doi:10.1007/s11067-015-9297-3, Online First

Harrod S (2011) Modeling network transition constraints with hypergraphs. Transp Sci 45(1):81–97

Higgins A, Kozan E (1998) Modeling train delays in urban networks. Transp Sci 32(4):346–357

Higgins A, Ferreira L, Kozan E (1996b) Optimisation of Train Schedules to Achieve Minimum Transit Times and Maximum Reliability, Proceedings of the 13th International Symposium on Transportation and Traffic Theory, 589–614

Higgins A, Kozan E, Ferreira L (1996b) Optimal scheduling of trains on a single-line track. Transp Res B 30(2):147–161

Higgins A, Kozan E, Ferreira L (1997) Heuristic techniques for single-line train scheduling. J Heuristics 3:43–62

Jong J-C, Chang S, Lai Y-C (2013) Development of a two-stage hybrid method for solving high speed rail train scheduling problem. Proceedings of the Transportation Research Board 92nd Annual Meeting, Washington D.C., January 13–17, 2013

Jovanovic D, Harker PT (1991) Tactical scheduling of rail operations: the SCAN I system. Transp Sci 25(1): 46–64

Karoonsoontawong A (2015) Efficient insertion heuristics for multi-trip vehicle routing problem with time windows and shift time limits. J Transp Res Board 2477:27–39

Karoonsoontawong A, Heebkhoksung K (2015) A modified wall building-based compound approach for knapsack container loading problem. Maejo Int J Sci Technol 9(1):93–107

Karoonsoontawong A, Lin D-Y (2015) Combined gravity model trip distribution and paired combinatorial logit stochastic user equilibrium problem. Netw Spat Econ. doi:10.1007/s11067-014-9279-x, Online First

Karoonsoontawong A, Unnikrishnan A (2013) Inventory routing problem with route duration limits and stochastic inventory capacity constraints: tabu search heuristics. J Transp Res Board 2378:43–53

Khan MB, Zhou X (2010) Stochastic optimization model and solution algorithm for robust double-track train-timetabling problem. IEEE Trans Intell Transp Syst 11(1):81–89

Kraay DR, Harker PT (1995) Real-time scheduling of freight railroads. Transp Res B 29:213–229

Lee Y, Chen C (2009) A heuristic for the train pathing and timetabling problem. Transp Res B 43(8–9):837–851

Li KP, Gao ZY, Mao BH, Cao CX (2011) Optimizing train network routing using deterministic search. Netw Spat Econ 11(2):193–205

Liu Q, Huang C (2015) Modeling business land use equilibrium for small firms' relocation and consumers' trip in a transportation network. Netw Spat Econ. doi:10.1007/s11067-015-9286-6, Online First

Ma J, Hu S, Xu H, Fan J (2002) A multi-objective model for train working diagram for Jinghu high-speed train line. Proceedings of the Third International Conference on Traffic and Transportation ICTTS, Beijing, 350–355

Meng L, Zhou X (2014) Simultaneous train rerouting and rescheduling on an N-track network: a model reformulation with network-based cumulative flow variables. Transp Res B 67:208–234

Meng L, Luan X, Zhou X (2015) A train dispatching model under a stochastic environment: stable train routing constraints and reformulation. Netw Spat Econ. doi:10.1007/s11067-015-9299-1, Online First

Mu S, Dessouky M (2011) Scheduling freight trains traveling on complex networks. Transp Res B 45:1103–1123

Newman AM, Nozick L, Yano CA (2002) Optimization in the rail industry. Handbook of Applied Optimization, pp.704-718

Petersen ER, Taylor AJ, Martland CD (1986) An introduction to computer aided train dispatching. J Adv Transp 20:63–72

Sahin I (1999) Railway traffic control and train scheduling based on inter-train conflict management. Transp Res B 33(7):511–534

Szpigel B (1973) Optimal train scheduling on a single track railway. Operations Research'72, North-Holland, Amsterdam, Netherlands, pp. 343–352

Törnquist J, Persson J (2007) A N-tracked railway traffic re-scheduling during disturbances. Transp Res B 41: 342–362

Xiao L, Lo HK (2014) Combined route choice and adaptive traffic control in a day-to-day dynamical system. Netw Spat Econ. doi:10.1007/s11067-014-9248-4, Online First

Zhou X, Zhong M (2005) Bicriteria train scheduling for high-speed passenger railroad planning applications. Eur J Oper Res 167:752–771

Zhou X, Zhong M (2007) Single-track train timetabling with guaranteed optimality: branch-and-bound algorithms with enhanced lower bounds. Transp Res B 41:320–341