A Lagrangian Relaxation Approach Based on a Time-Space-State Network for Railway Crew Scheduling

Ying Wang[a], Zheming Zhang[b], Dennis Huisman[c,d], Andrea D'Ariano[e], Jinchuan Zhang[a,*]

[a] *School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China*

[b] *Hangzhou Pi-Solution Information Technology Co., Ltd, Hangzhou 311121, China*

[c] *Econometric Institute and Erasmus Center for Optimization in Public Transport, Erasmus University Rotterdam, P.O. Box 1738, NL-3000 DR Rotterdam, The Netherlands*

[d] *Process Quality and Innovation, Netherlands Railways, P.O. Box 2025, NL-3500 HA Utrecht, The Netherlands*

[e] *Department of Engineering, Roma Tre University, Rome 00146, Italy*

Econometric Institute Report EI2018-45

Abstract: The crew scheduling problem is an important and difficult problem in railway crew management. In this paper, we focus on the railway crew scheduling problem with time window constraints caused by meal break rules. To solve this optimization problem, a solution method is proposed based on a time-space-state network and Lagrangian relaxation. In this method, the "hard constraints" corresponding to the crew rules are described as the state of vertices in the time-space-state network. Based on the network, this problem is modeled as a network flow model, referred to as an initial model. To break the symmetry and improve the strength of the formulation, five valid inequalities are added. To solve the problem, we relax the coupling constraints by Lagrangian relaxation. The resulting subproblems are shortest path problems in the time-space-state networks. We propose a Lagrangian heuristic to find a feasible solution. Finally, the solution method is tested on real-world instances from an intercity rail line and a regional railway network in China. We discuss the effects of additional valid inequalities and the effects of different length of meal time windows.

Keywords: Scheduling; Meal time window; Time-space-state network; Lagrangian relaxation

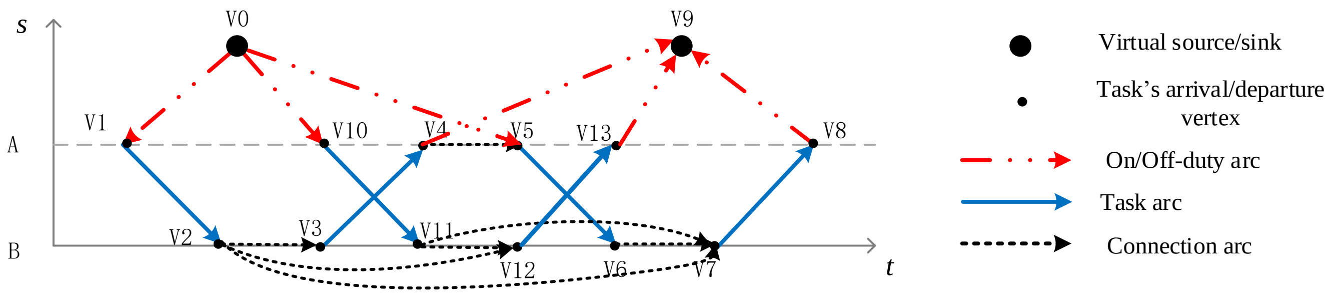
*Corresponding author.
E-mail address: jchzhang@bjtu.edu.cn (J.C. Zhang).

1 Introduction

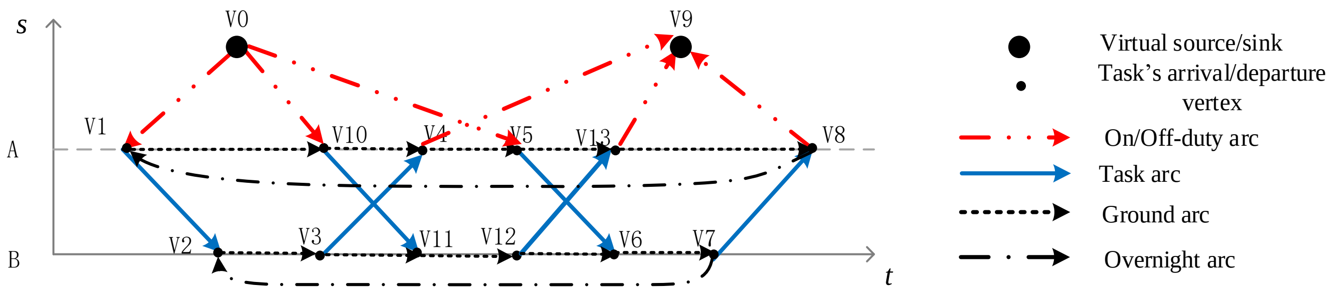
Crew management is an important aspect of railway operations, while the crew scheduling problem (CSP) is a core problem in crew management. This paper focuses on the crew scheduling problem with Chinese meal break constraints. In many papers, meal break constraints set limits on the times of having meals and the time length of meal break. However, in Chinese railway operation, the meal break constraints not only set limits on these two aspects but also set limits on the time window to have a meal. For a day duty that lasts from morning to afternoon, there must be a meal break for lunch in a given time window. For the lunch break, the time window usually starts at 11:00 am and ends at 1:00 pm. While for a night duty that lasts from afternoon to evening, there must be a meal break for supper in a given time window usually between 5:00 pm to 7:00 pm.

In the CSP, most of the crew rules constrain accumulated time of some activities, the maximum working time of a duty and maximum consecutive driving time. In contrary to the crew rules on the duration of certain activities, the meal break rules constrain the start and end moments of certain activities. The former constrains on "relative time", while the latter constrains on "absolute time". In this paper, we call the former type of constraints "duration constraints", the latter type "moment constraints" and both of them "mixed time constraints". Obviously, the CSP is an optimization problem with "mixed time constraints". In this paper, we apply a time-space-state network to describe the "mixed time constraints".

The CSP is usually modeled as a set covering model based on a connection network. Sherali (2006) compared two typical network constructions: connection network and time-space network. The former uses the arcs to represent feasible connections between train tasks, while the latter focuses on representing train tasks and leaves to the model the decision on the connections, as long as these are feasible due to time and space considerations. Fig. 1 gives an example of the same problem while constructing different types of networks.



(a) Example of a connection network



(b) Example of a time-space network

Fig. 1 Example of different types of networks to represent the same crew scheduling problem

In Fig. 1, A and B represent two stations. Station A is the station where the crew depot is located. Vertices v_0 and v_9 represent the virtual source and sink corresponding to the same crew depot. There are six tasks modeled as task arcs. For the departure/arrival of each task, there is a departure/arrival vertex in both types of networks. The difference of two types of networks lies on the connection arcs and ground arcs. In Fig. 1(a), there is a connection arc for each feasible connection. For example, there are three feasible connections from vertex v_2 to v_3, v_{11} and v_7. Therefore, it can easily grow to an unmanageable size due to the large number of possible connections. In Fig. 1(b), along the time line of each station, there is a ground arc between every two nearby vertices, e.g. arc $(v_1,\ v_{10})$. There is only an inbound arc for each departure vertex and two outbound arcs for each arrival vertex. Therefore, there is a limited number of vertices and arcs in the time-space network. Overall, the connection network is more powerful on managing the "duration constraints", but has as a disadvantage of a large number of arcs in the network. On the other hand, the time-space network is more suitable to deal with "moment constraints", except complex constraints in the model.

To combine the advantages of a connection network and a time-space network, this paper adopts the time-space-state network (TSSN) of Mahmoudi and Zhou (2016), who proposed a TSSN based approach to solve a vehicle routing problem with pickup and delivery time windows. TSSN is a three-dimensional network, which adds the dimension "state" into a two-dimensional time-space network. Each state of a vertex in TSSN represents a feasible status from the source vertex to the current vertex satisfying routing feasibility constraints. Thus, all constraints on routings, except the flow balance constraints, are described as the feasible status of vertices in TSSN and in the optimization model. There are only constraints on the coupling of routings and flow balance. The main idea of the model based on TSSN is to move complex constraints on routings from the optimization model into the network construction so that the vehicle routing problem can be solved by some classical optimization methods. In particular, the routing generation in TSSN is transformed into the classical shortest path problem that can be solved in polynomial time. For CSP, we use a state dimension to describe feasible states of each task vertex according to "mixed time constraints" (The details of constructing the TSSN will be discussed later). This network is very suitable for a Lagrangian relaxation approach. Therefore, we will develop a Lagrangian relaxation approach to solve the CSP.

The contribution of this paper is as follows. First, we apply TSSN to represent "mixed time constraints" of CSP. All crew rules of the CSP are described in the TSSN, and the CSP can be modeled as a network flow problem. Secondly, we present two strategies to break the symmetry and to strengthen the initial network flow model based on TSSN (these two strategies can also be applied to solve similar problems). Thirdly, we develop a Lagrangian heuristic to solve the improved model. Finally, we investigate the benefits of our approach by using real-world instances and discuss the performance of our approach on different types of instances.

The remainder of this paper is organized as follows. In Section 2, we give an overview of related work. In Section 3, we describe the CSP. In Section 4, we introduce the definition of the "state" dimension of TSSN and the construction method of the TSSN. In Section 5, we propose an initial network flow model based on TSSN and add five valid inequalities to break the symmetry and to improve the strength of the initial model. In Sections 6, we introduce our Lagrangian heuristic, and we apply this approach to real-world instances in Section 7. Finally, this paper gives conclusions and practical recommendations in Section 8.

2 Literature Review

Much valuable research work on CSP has been done on airline (see the surveys of Barnhart et al. 2003; Ernst et al. 2004; Gopalakrishnan and Johnson 2005), railway (Caprara et al. 1997, 2007; Abbink et al. 2018) and urban mass transit (see Desrochers et al. 1989; Lourenço et al. 2001; Tallys et al. 2005; Huisman et al. 2005; Chen Shijun et al. 2013). Many articles devoted to methodologies and applications in airline crew scheduling are earlier than to any other application area. An earlier survey, in 1969, is given by Arabeyre et al. Kasirzadeh et al. (2017) reviewed existing problem formulations and solution methodologies of the airline crew scheduling problem and concluded that the most popular approach since the 1990s is the set covering problem with column generation embedded in a branch-and-bound method.

Because of the much higher combinatorial explosion of the railway CSP than the airline CSP, the application of these models in the railway industry is prohibitive until the late 1990s. The railway CSP is also usually modeled as a set covering problem with additional constraints (Caprara et al. 1997, 1999; Freling et al. 2000; Abbink et al.2011; Nishi et al. 2011; Wang et al. 2009) or a set partitioning problem (Kohl 2003) based on connection networks which are flexible and powerful when generating feasible duties. Much effort is paid on developing efficient algorithms to solve the large-scale set covering/partitioning models according to the particular features of the railway CSP. The three most common solution methodologies are branch-and-price (Freling et al. 2000; Wang et al. 2009), Lagrangian relaxation (Caprara et al. 1997, 2001) and a combination of column generation with a decomposition algorithm, heuristic algorithm or relaxation algorithms (Kohl 2003; Abbink et al. 2011; Nishi et al. 2011; Huisman 2007; Jütte et al. 2015). No matter which solution method is applied, research is mainly devoted to exploring more effective strategies to improve the solution quality and efficiency, such as faster strategies to generate feasible duties, strategies to reduce the network size, more effective branching/pruning strategies, duty selection strategies. Besides the set covering model, Vaidyanathan et al. (2007) and Sahin et al. (2011) also formulated the CSP as a multi-commodity flow model and a network flow model based on a space-time network, respectively. Both of the two network flow models are solved by heuristics.

As summarized above, decomposition methods such as column generation are popular to solve the railway CSP because of its inherent complex constraints. In order to describe the "mixed time constraints", we utilize TSSN to formulate the railway CSP. Based on TSSN, the "duty generation" can be transformed into a classic shortest path problem. The TSSN provides a good general framework to embed different algorithms, such as Lagrangian Relaxation, Column Generation and heuristic methods. Even though the search space created by TSSN has multiple dimensions and is accordingly large in its size, the polynomial algorithm of the shortest path problem and a large amount of computer memory and power in modern workstations can accommodate the multi-dimensional solution vectors. In this paper, we use Lagrangian relaxation to solve the CSP.

3 Problem Description

3.1 Terminology

Depot: Each crew member belongs to one crew *depot*, where each of his/her duties starts and ends.

Task: Each train service has been split into a sequence of *tasks*, defined as segments of train journeys which must be served by the same crew without rest. Each task is characterized by departure time, a departure station, an arrival time, an arrival station, and possibly additional attributes.

Duty: Each *duty* represents a sequence of consecutive *tasks* to be carried out by a single crew member within a single day. Within each duty, two consecutive tasks end and start at the same station. Specifically, the second task starts later than completion of the first task. A duty is a single workday for one anonymous crew member that is sandwiched between two overnight rest periods.

Pairing: Each *pairing* starts and ends at the same depot. Each pairing also represents a sequence of consecutive *tasks* to be carried out by a single crew member within several days depending on the train timetable. In most situations, a duty is also a pairing, given that the duty starts and ends at the same depot. However, for some night train services, the period of a pairing could be two (or more) days. In this case, the pairing contains multiple duties.

Crew Schedule: A crew schedule is a set of anonymous *pairings* that covers all given tasks exactly once (sometimes more than once depending on whether deadheading is allowed).

The railway CSP can be defined shortly as follows: Given a set of tasks, find a feasible crew schedule with minimal costs such that all tasks are covered. In this context feasible means that all crew rules are satisfied.

Table 1 shows an example of crew schedule, according to the rolling stock circulation showed in Fig. 2. The crew can transfer at stations B and C and have an overnight rest at station C. G1, G2, G3-1, ..., G6 are the tasks. G3-1and G3-3 belong to the same train service G3 while G4-2 and G4-4 belong to the same train service G4. G3 and G4 are operated by the same train units, while G1, G2, G5, and G6 are operated by other train units. In Table 1, pairings J1 and J3 are both single duties, while pairing J2 contains two duties.

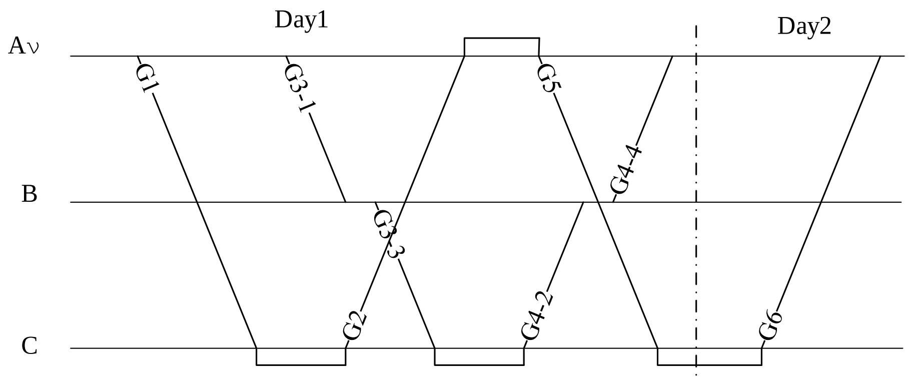


Fig. 2 Example of rolling stock circulation

Table 1 Example of crew schedule with meal break and overnight rest

No. of crew pairing	Contents of crew pairing
J1	G1→meal break→G2
J2	G5→ overnight rest →G6
J3	G3-1→G3-3→ meal break→G4-2→G4-4

3.2 Crew rules

We consider the following crew rules in our problem.

(1) Working time of a duty: the accumulated working hours (including driving, break, transfer) Td of a duty must not be longer than the maximum working time of a duty Td^{max}.

(2) Transfer time: when the crew consecutively carries out tasks operated by different train units, the time interval Tt between the arrival time of the first task and the departure time of the following task must not be shorter than the minimum transfer time Tt_{min}.

(3) Consecutive driving time: when the crew consecutively carries out tasks (including transfer time but not break time), the consecutive driving time To must be no longer than the maximum consecutive driving time To^{max}.

(4) Break time: After carrying out consecutive tasks, the crew must have a break time for a short rest (or meal). The break time Tr must be not shorter than the minimum break time Tr_{min}.

(5) Overnight rest time: when a crew member needs to have an overnight rest out of his/her depot, the rest time Ts

should be at least overnight rest time Ts_{min}.

(6) The period of a pairing: the crew must return to his/her depot within Dd days which means that the time interval between the beginning and end of a pairing Tp must not be longer than Dd days.

(7) Meal break time: the crew must have meals in the given time window $TW_{MB}^{l} = [ML_{min}, ML_{max}]$ and $TW_{MB}^{s} = [MS_{min}, MS_{max}]$ for lunch and supper, respectively. The dining time Te must not be shorter than the minimum meal time Te_{min}. In addition, the meal break has to be Te_{b}^{a} hours after the start of the duty and Te_{f}^{b} hours before the end of the duty.

In the above seven crew rules, the first six crew rules are all “relative time” – “duration” constraints, while the time window of meals is “absolute time” – “moment” constraint. Therefore, from the perspective of the time dimension attribute of the constraints, the CSP is a large-scale combinatorial optimization problem with "mixed time constraints".

3.3 Discussion of meal break constraints

As mentioned in Section 3.2, the crew must have their meals in a given time window. In this paper, if the start of a duty is at least the minimum meal time after the start of a meal time window (e.g. lunch), we assume that each crew has a meal (lunch) before the duty. For example, suppose that the minimum meal time is 30 minutes and the lunch time window is [11:00, 13:00]. If a duty begins at 11:30 am, then we assume that each crew has had lunch before he/she carries the duty and we do not consider the lunch break constraint for the rest of his/her duties. Similarly, if the end of a duty is at least the minimum meal time before the end of a meal time window, we assume that each crew has a meal (lunch) after the duty. Therefore, only when the duration of a duty covers most part of a meal time window and the uncovered time is shorter than the minimum meal time, the given meal break constraints need to be considered in the duty generation.

The meal break is usually arranged when each crew has a short rest. In this paper, when the break time window overlaps with the meal time window and the overlap time is not shorter than the minimum meal time, we assume that each crew has a meal during the break.

4. Construction of the time-space-state network

4.1 Definition of "state" dimension of TSSN

In the time-space network, each vertex (t, s) has two dimensions, time and space. Vertex (t, s) and vertex (t, s’) are two different vertices in a time-space network with the same time dimensional value while with different space dimensional values. In the TSSN, each vertex has three dimensions (t, s, ω), in which ω represents the state dimension.

We define the state of the vertex v_i by means of five elements according to crew rules. The set of all states of vertex v_i is defined as $\Omega_i = \{\omega_{i(1)}, \omega_{i(2)}, \dots, \omega_{i(m)}\}$. We also define a time-space-state vertex $(t_i, s_i, \omega_{i(m)})$ as $v_{i(m)}$. Accordingly, the five attributes of state of $v_{i(m)}$ are defined as $\omega_{i(m)} = (Td_{i(m)}, To_{i(m)}, Tc_{i(m)}, Tp_{i(m)}, Mb_{i(m)})$. The meaning and calculation of each element are explained as follows:

-$Td_{i(m)}$ represents the accumulated working time of vertex $v_{i(m)}$ in a duty. The value of $Td_{i(m)}$ is calculated according to its predecessor vertex. We assume that $v_{j(n)}$ is the predecessor vertex of $v_{i(m)}$, which means that there is an arc from $v_{j(n)}$ to $v_{i(m)}$. We define the length of the arc ($v_{j(n)}$, $v_{i(m)}$) as $tt_{j(n),i(m)} = t_i - t_j$. If $v_{j(n)}$ is the depot, the value of $Td_{i(m)}$ will be set to zero which means that a duty will start. If arc ($v_{j(n)}$, $v_{i(m)}$) is an overnight arc, the value of $Td_{i(m)}$ will also be set to zero which means that a duty is over and crew will have an overnight rest.

Otherwise, $Td_{i(m)} = Td_{j(n)} + tt_{j(n),i(m)}$.

-$To_{i(m)}$ represents the accumulated consecutive driving time of vertex $v_{i(m)}$ in a duty. The value of $To_{i(m)}$ is calculated according to its predecessor vertex $v_{j(n)}$. Similarly to $Td_{i(m)}$, if $v_{j(n)}$ is the depot or arc ($v_{j(n)}$, $v_{i(m)}$) is an overnight arc, the value of $To_{i(m)}$ will be set to zero. Moreover, if there is a short rest or meal break on arc ($v_{j(n)}$, $v_{i(m)}$), the value of $To_{i(m)}$ should also be set to zero which means that consecutive driving is over. Otherwise, $To_{i(m)} = To_{j(n)} + tt_{j(n),i(m)}$.

-$Tc_{i(m)}$ represents the accumulated break/waiting time of vertex $v_{i(m)}$ in a duty. The value of $Tc_{i(m)}$ is calculated according to its predecessor vertex $v_{j(n)}$. If arc ($v_{j(n)}$, $v_{i(m)}$) is a task arc, the value of $Tc_{i(m)}$ will be set to zero. Otherwise, $Tc_{i(m)} = Tc_{j(n)} + tt_{j(n),i(m)}$.

-$Tp_{i(m)}$ represents the accumulated working time of vertex $v_{i(m)}$ in a pairing. The calculation of $Tp_{i(m)}$ is the same as $Td_{i(m)}$ except that when arc ($v_{j(n)}$, $v_{i(m)}$) is an overnight arc, the value of $Tp_{i(m)}$ should not be set to zero, but it can still be calculated as $Tp_{i(m)} = Tp_{j(n)} + tt_{j(n),i(m)}$.

-$Mb_{i(m)}$ represents the dining status for lunch and supper. This value is used to determine whether each crew has lunch or supper in the given time windows. If the accumulated consecutive driving time is no less than the minimum working hours before a meal break, the accumulated break/waiting time is no less than the minimum meal time, and the current time of task is larger than the start of lunch (or supper) time window at least the minimum meal time, that is $To_{i(m)} \geq Te_b^a$, $Tc_{i(m)} \geq Te_{min}$ and $t_i - ML_{min} \geq Te_{min}$ (or $t_i - MS_{min} \geq Te_{min}$), there is a meal break for lunch (or supper). We here define three values of dinning status to indicate before lunch, after lunch but before supper, after supper.

We provide an example of calculating these five elements in Section 4.2.2.

4.2 Construction of TSSN

4.2.1 Constructing basic time-space network (BTSN)

Given the set of tasks, we construct the basic time-space network $G_{ts} = (V_{ts}, A_{ts})$. V_{ts} represents the set of all vertices in the network G_{ts}, while A_{ts} represents the set of all arcs. We define V_{ts}^o, V_{ts}^l and V_{ts}^d as the set of virtual source vertices, the set of tasks' arrival and departure vertices and the set of virtual sink vertices, respectively. Then, there is $V_{ts} = V_{ts}^o \cup V_{ts}^l \cup V_{ts}^d$. Similarly, we define A_{ts}^o, A_{ts}^l, A_{ts}^g, A_{ts}^{on} and, A_{ts}^d as the set of on-duty arcs, the set of task arcs, the set of ground arcs, the set of overnight arcs and the set of off-duty arcs, respectively. In addition, we add a virtual arc between each pair of source and sink vertices related to the same depot, in order to represent the crew staying at this depot without carrying out tasks. The set of virtual arcs is defined as A_{ts}^{od}. We can easily show that $A_{ts}^{od} \in A_{ts}^o$, $A_{ts}^{od} \in A_{ts}^d$ and $A_{ts} = A_{ts}^o \cup A_{ts}^l \cup A_{ts}^g \cup A_{ts}^{on} \cup A_{ts}^d$. Each virtual arc is a pairing covering no tasks. We call this kind of pairing as staying pairing which means that the crew stay at a depot and do not carry out tasks.

Vertex v_i has four attributes ($t_i, s_i, \varphi_i, \theta_i$) representing the arrival/departure time of the task φ_i, the arrival/departure station of task φ_i, the index of a corresponding task and the index of train units routing covering task φ_i, respectively. For all $o \in V_{ts}^o$ and $d \in V_{ts}^d$, vertex o and vertex d have attributes s_o and s_d representing the index of a station, which is also a crew depot. For all $(i,j) \in A_{ts}$, arc (i,j) has two attributes (tt_{ij}, θ_{ij}) representing the length and the type of this arc (such as on-duty arc, task arc, ground arc, etc.). For all $(i,j) \in A_{ts}^o \cup A_{ts}^d \backslash A_{ts}^{od}$, $tt_{ij} = 0$; for all $(i,j) \in A_{ts}^l \cup A_{ts}^g$, $tt_{ij} = t_j - t_i$; for all $(i,j) \in A_{ts}^{on}$, $tt_{ij} = t_j - t_i$+1440; for all $(i,j) \in A_{ts}^{od}$, $tt_{ij} = tt_{od}$.

The value of tt_{od} is defined as an integer number larger than 0 and much less than Td^{max}.

4.2.2 Generation of state dimension

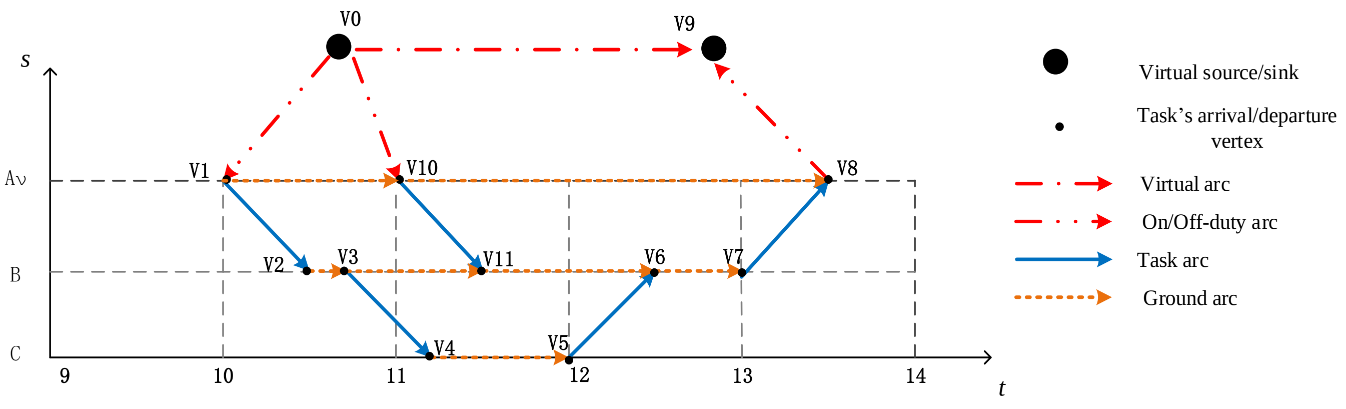
The state dimension of BTSN is generated by using a breadth-first search algorithm that is shown in Appendix B. The new generated network is a TSSN represented by $G_{tss} = (V_{tss}, A_{tss})$. We also define V_{ts}^{o}, V_{tss}^{l}, V_{tss}^{d}, A_{tss}^{o}, A_{tss}^{l}, A_{tss}^{g}, A_{tss}^{on}, A_{tss}^{d} and A_{tss}^{od} with the same meaning as in BTSN.

Based on the construction of TSSN, this network has two main characteristics.

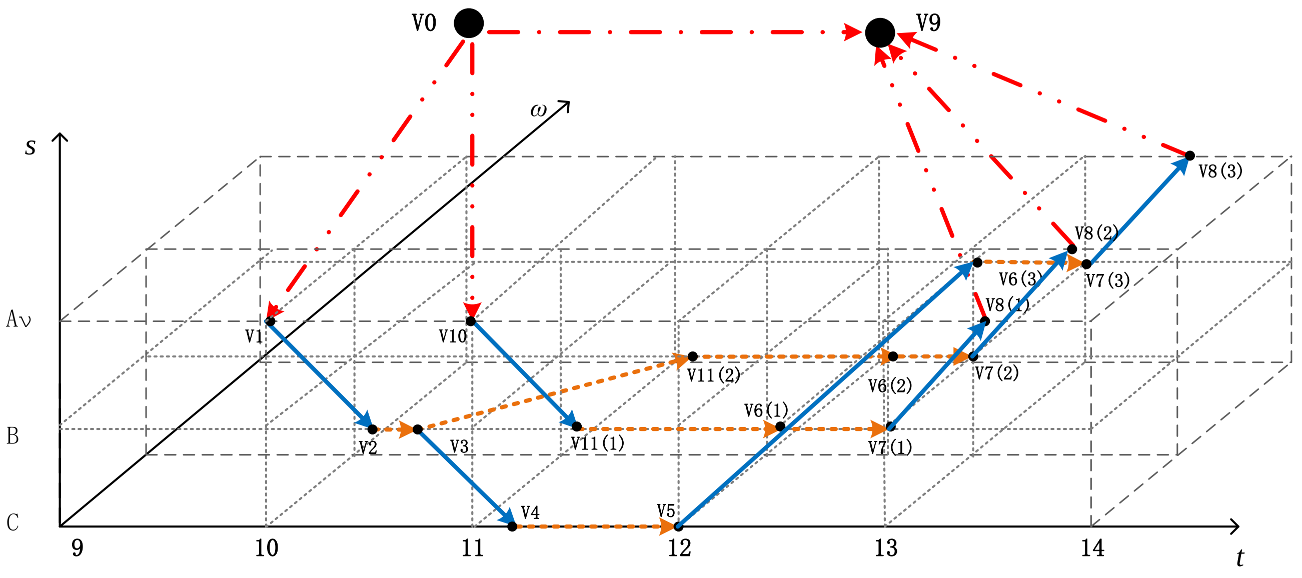
- The TSSN is a topologically ordered directed graph. Since the TSSN is generated based on the train schedule, all arcs in the TSSN are in the same direction as the time line and the start time is smaller than the end time of the arc.

- The TSSN is a sparse graph. For each vertex, except the virtual source and sink, there are at most two inbound arcs, a task arc and a(n) ground/overnight arc, and at most two outbound arcs, a task arc and a(n) ground/overnight arc. Therefore, for each vertex, except the virtual source and sink, there is one inbound arc and at most two outbound arcs. During the construction of TSSN, only when the newly generated state dimensional value $\omega_{i(m)}$ of a vertex v_i is the same as its existing state dimensional value (supposed to be $\omega_{i(m-1)}$), a new arc will be connected to an existing vertex $v_{i(m-1)}$, and thus the vertex $v_{i(m-1)}$ would have more than one inbound arcs.

Fig. 3 shows an example of constructing TSSN from BTSN.



(a) Basic time-space network



(b) Time-space-state network

Fig. 3 Comparison between the time-space network and time-space-state network

Fig. 3(a) is an example of the basic time-space network of the railway CSP similar to Fig. 1(b). For clarity reasons, we omit overnight connection arcs in Fig. 3(a). Fig. 3(b) is the constructed time-space-state network of Fig. 3(a). For simplicity reasons, we set both of the minimal short break time and meal break time to 45 minutes. The time window for lunch and supper are [11:00, 13:00] and [17:00, 19:00], respectively. Then the three-dimensional values of the node $v_{i(m)}$ are given in Table 2.

Table 2 Three-dimensional value of the node $v_{i(m)}$ in Fig. 3(b)

$v_{i(m)}$	$t_{i(m)}$	$s_{i(m)}$	$\omega_{i(m)}$				
			$Td_{i(m)}$	$To_{i(m)}$	$Tc_{i(m)}$	$Tp_{i(m)}$	$Mb_{i(m)}$
v_1	10:00	A	0	0	0	0	0
v_2	10:30	B	30	30	0	30	0
v_3	10:45	B	45	45	15	45	0
v_4	11:15	C	75	75	0	75	0
v_5	12:00	C	120	0	45	120	1
$v_{6(1)}$	12:30	B	90	0	60	90	1
$v_{6(2)}$	12:30	B	150	0	105	150	1
$v_{6(3)}$	12:30	B	150	30	0	150	1
$v_{7(1)}$	13:00	B	120	0	90	120	1
$v_{7(2)}$	13:00	B	180	0	135	180	1
$v_{7(3)}$	13:00	B	180	60	30	180	1
$v_{8(1)}$	13:30	A	150	30	0	150	1
$v_{8(2)}$	13:30	A	210	30	0	210	1
$v_{8(3)}$	13:30	A	210	90	0	210	1
v_{10}	11:00	A	0	0	0	0	0
$v_{11(1)}$	11:30	B	30	30	0	30	0
$v_{11(2)}$	11:30	B	90	0	45	90	0

In Table 2, $t_{i(m)}$ represents the departure/arrival time of vertex $v_{i(m)}$, while $s_{i(m)}$ represents the departure/arrival station of vertex $v_{i(m)}$. Taking path (v_0-v_{10}-$v_{11(1)}$-$v_{6(1)}$-$v_{7(1)}$-$v_{8(1)}$-v_9) in Fig. 3(b) as an example, we explain the calculation of $\omega_{i(m)}$. Regarding vertex v_{10}, its predecessor is vertex v_0 which is a virtual source. The value of Td_{10}, To_{10}, Tc_{10} and Tp_{10} are all zero. For t_{10} equal to 11:00, which is just the same as the start of the time window for lunch, the value of Mb_{10} is zero. Regarding vertex $v_{11(1)}$, its predecessor is vertex v_{10} and the length of the arc (v_{10},

v_{11}) is 30 minutes. The values of $Td_{11(1)}$, $To_{11(1)}$ and $Tp_{11(1)}$ are 30 minutes. If (v_{10}, v_{11}) is a task arc, the values of $Tc_{11(1)}$ and $Mb_{11(1)}$ are both zero. Regarding vertex $v_{6(1)}$, its predecessor is vertex $v_{11(1)}$ and the length of arc (v_{11}, v_6) is 60 minutes, which is a ground arc. The values of $Td_{6(1)}$ and $Tp_{6(1)}$ are both 90 minutes, the value of $To_{6(1)}$ is zero and the value of $Tc_{6(1)}$ is 60 minutes. Arc (v_{11}, v_6) is a ground arc and its length is larger than the short break time and the meal break time. The value of $Mb_{6(1)}$ is 1, which means each crew member has lunch during the short break. For vertex $v_{7(1)}$, its predecessor is vertex $v_{6(1)}$ and the length of arc (v_6, v_7) is 30 minutes, which is a ground arc. The value of $Td_{7(1)}$ and $Tp_{7(1)}$ are both 120 minutes, the value of $To_{7(1)}$ equals to the value of $To_{6(1)}$, the value of $Tc_{7(1)}$ is 90 minutes and the value of $Mb_{7(1)}$ equals to the value of $Mb_{6(1)}$. For vertex $v_{8(1)}$, its predecessor is vertex $v_{7(1)}$ and the length of arc (v_7, v_8) is 30 minutes, which is a task arc. The value of $Td_{8(1)}$ and $Tp_{8(1)}$ are both 150 minutes. The value of $To_{8(1)}$ is 30 minutes, the value of $Tc_{8(1)}$ is zero and the value of $Mb_{8(1)}$ equals to the value of $Mb_{7(1)}$.

5 Optimization model for the crew scheduling problem

5.1 Initial model

In the railway CSP, each duty is carried out by a driver or a conductor team on a single day. The number of crew duties implies the number of crew members needed, which should be as few as possible. Moreover, crew members prefer to go on duty early/late and go off duty early/late for the same driving duration, so that they could have more rest time at home rather than at station. Therefore, we minimize the sum of the working time of each crew member. The TSSN-based optimization model (M1) of the railway CSP is formulated as follows.

M1: $$Z = \min \sum_{p \in P} \sum_{(i,j) \in A_{tss}} tt_{ij} x_{ij}^p \tag{1}$$

$$\sum_{p \in P} \sum_{(i,j) \in A_{tss}^l, \varphi_i = \varphi_j = l} x_{ij}^p \geq 1 \quad \forall l \in L \tag{2}$$

$$\sum_{(i,j) \in A_{tss}} x_{ij}^p = \sum_{(j,i) \in A_{tss}} x_{ji}^p \quad \forall i \in V_{tss}^l, p \in P \tag{3}$$

$$\sum_{(i,j) \in A_{tss}^o} x_{ij}^p = 1 \quad \forall p \in P \tag{4}$$

$$x_{ij}^p \in \{0,1\} \quad \forall (i,j) \in A_{tss}, p \in P \tag{5}$$

In M1, *L* is the set of tasks and *P* is the set of available crew members. The binary variable x_{ij}^p indicates whether the arc (i, j) is covered by crew member *p*. The objective function (1) minimizes the total working time (including staying time of virtual arcs). Constraint (2) ensures that each task must be covered by at least one crew member. Constraint (3) is the flow balance constraint. Constraint (4) ensures that each crew member either carries out a pairing or covers a virtual arc, which means staying at a depot and not covering any tasks. Because each task must be carried out by at least one crew member. The more crew members that cover virtual arcs, the less crew members are needed to carry out the pairings. As the length of virtual arcs is much less than Td^{max}, the objective function can ensure the minimization of the number of crew members needed and the crew preference for tight duties. Note that the size of set *P* should be set properly. If this value is too big, the problem size will be very large. Otherwise, no feasible solution is available.

5.2 Valid inequalities of "Breaking the symmetry" and "Strengthening the formulation"

5.2.1 Valid inequality of partly breaking the symmetry

In our model, the same pairing carried by different crew members is considered as different decision variables. However, the pairings are anonymous in the railway CSP, so it makes no sense that a pairing is carried out by different crew members. Therefore, symmetry should be avoided when solving the model. Because of the existence of symmetry, computation time is often wasted in finding new solutions, which are symmetric with respect to the already visited solutions. Hoffmann and Buscher (2018) avoided symmetry in their arc flow model by demanding that conductors were employed as the sequence of duties sorted in decreasing order according to their duty times. They added two inequalities, with the same meanings as the Constraints (6) and (7), into their model.

$$\sum_{(i,j)\in A_{tss}^{o}} x_{ij}^{p} \geq \sum_{(i,j)\in A_{tss}^{o}} x_{ij}^{p+1} \quad \forall p \in \{1,2,\ldots,|P|-1\} \tag{6}$$

$$\sum_{(i,j)\in A_{tss}} tt_{ij} x_{ij}^{p} \geq \sum_{(i,j)\in A_{tss}} tt_{ij} x_{ij}^{p+1} \quad \forall p \in \{1,2,\ldots,|P|-1\} \tag{7}$$

These two inequality constraints indeed strengthen the formulation. However, the total working time of two crew members can be the same in Constraint (7), the symmetry of the model is not completely broken. Therefore, we propose various inequalities to completely break the symmetry of M1. Firstly, we add the following constraint to partly break symmetry.

$$\sum_{(i,j)\in A_{tss}^{o}} C_{o}^{p} x_{ij}^{p} + \sum_{(i,j)\in A_{tss}^{od}} C_{od}^{p} x_{ij}^{p} \geq \sum_{(i,j)\in A_{tss}^{o}} C_{o}^{p-1} x_{ij}^{p-1} + \sum_{(i,j)\in A_{tss}^{od}} C_{od}^{p-1} x_{ij}^{p-1} \quad \forall p \in P\setminus\{1\} \tag{8}$$

C_o^p is defined as an "on-duty cost" and C_{od}^p is defined as a "staying cost" for crew member p. We set $C_o^p > C_o^{p-1}$ and $\min C_o^p > \max C_{od}^p$. The constraint (8) ensures that the crew member carry out pairings as the sequence 1, 2, 3, …, |*P*|. If there are *p* crew members needed in a solution, then the first *p* crew members carry out pairings, while the crew members *p*+1 to |*P*| cover only virtual arcs, i.e. they stay at the depot. This constraint avoids wasting computation time on searching solution symmetry: the same number of virtual arcs are covered by random crew rather than the last |*P*| - *p* crew members. The model with constraint (8) can be formulated as M2.

M2: $$Z = \min \sum_{p\in P} \sum_{(i,j)\in A_{tss}} tt_{ij} x_{ij}^{p}$$

St. Constraints (2) - (5), (8)

5.2.2 Valid inequalities of further breaking the symmetry and strengthening the formulation

In M1, only covering constraint (2) constrains the coupling of pairings. Constraint (3) and (4) both constrain the feasibility of a single pairing. Due to the special structure of M1, it is a good choice to use the Lagrangian relaxation to relax the coupling constraint (2) as a penalty in the objective function. Then, the problem is decomposed into |*P*| independent shortest paths in TSSN during each iteration. However, because of symmetry, this will cause that the pairing with shortest working time is carried out by every crew member when using Lagrangian relaxation to get a lower bound solution. To strengthen the formulation, we further add the following additional constraints to M2. This addition would increase the coupling of pairings carried out by two different crew members.

$$\sum_{(i,j)\in A_{tss}^{l}} e_{ij}^{pq} + \sum_{(i,j)\in A_{tss}^{od}} (x_{ij}^{p} + x_{ij}^{q}) \geq 1 \quad \forall p \in P\setminus\{1\}, q \in P\setminus\{|P|\}, p > q \tag{9}$$

$$2e_{ij}^{pq} \geq 2-(x_{ij}^{p}+x_{ij}^{q}) \quad \forall (i,j)\in A_{tss}^{l}, p\in P\setminus\{1\}, q\in P\setminus\{|P|\}, p>q \tag{10}$$

$$e_{ij}^{pq} \leq 2-(x_{ij}^{p}+x_{ij}^{q}) \quad \forall (i,j)\in A_{tss}^{l}, p\in P\setminus\{1\}, q\in P\setminus\{|P|\}, p>q \tag{11}$$

$$e_{ij}^{pq} \in \{0,1\} \quad \forall (i,j)\in A_{tss}^{l}, p\in P\setminus\{1\}, q\in P\setminus\{|P|\}, p>q \tag{12}$$

The newly added auxiliary decision variable e_{ij}^{pq} represents whether crew pairing p and q cover task arc (i,j) at the same time. If crew pairing p and q do not cover task arc (i,j) at the same time, the value of e_{ij}^{pq} is 1; Otherwise, $e_{ij}^{pq}=0$. This is ensured by Constraints (10) to (12). Constraint (9) ensures that the pairing carried out by the latter crew member cannot be completely the same with the pairing carried out by the former crew member. Thus, the symmetry of the M2 can be totally broken by the added inequalities and formulation can also be strengthened. The final improved model can be formulated as M3.

M3: $$Z=\min \sum_{p\in P}\sum_{(i,j)\in A_{tss}} tt_{ij}x_{ij}^{p}$$

St. Constraints (2) - (5), (8)-(12)

6 Lagrangian heuristic

6.1 Lagrangian dual model

In M3, the strong coupling constraint (2) is relaxed into the objective function (13) by using Lagrangian multipliers as penalties. The Lagrangian dual model is shown as M4.

M4: $$LD=\max_{\rho\geq 0} L(\rho) \tag{13}$$

$$L(\rho)=\min \sum_{p\in P}\sum_{(i,j)\in A_{tss}} tt_{ij}x_{ij}^{p}+\sum_{l\in L}\rho(l)(1-\sum_{p\in P}\sum_{(i,j)\in A_{tss}^{l},\varphi_i=\varphi_j=l} x_{ij}^{p}) \tag{14}$$

St. Constraints (3) - (5), (8) - (12)

In Constraint (14), $\rho(l)$ represents the Lagrange multiplier of the task l, indicating the penalty value when the solution obtained does not satisfy the constraint (2).

We propose a Lagrangian relaxation-based heuristic to solve the improved model (M3). An overview of the Lagrangian heuristic is outlined in Fig. 4.

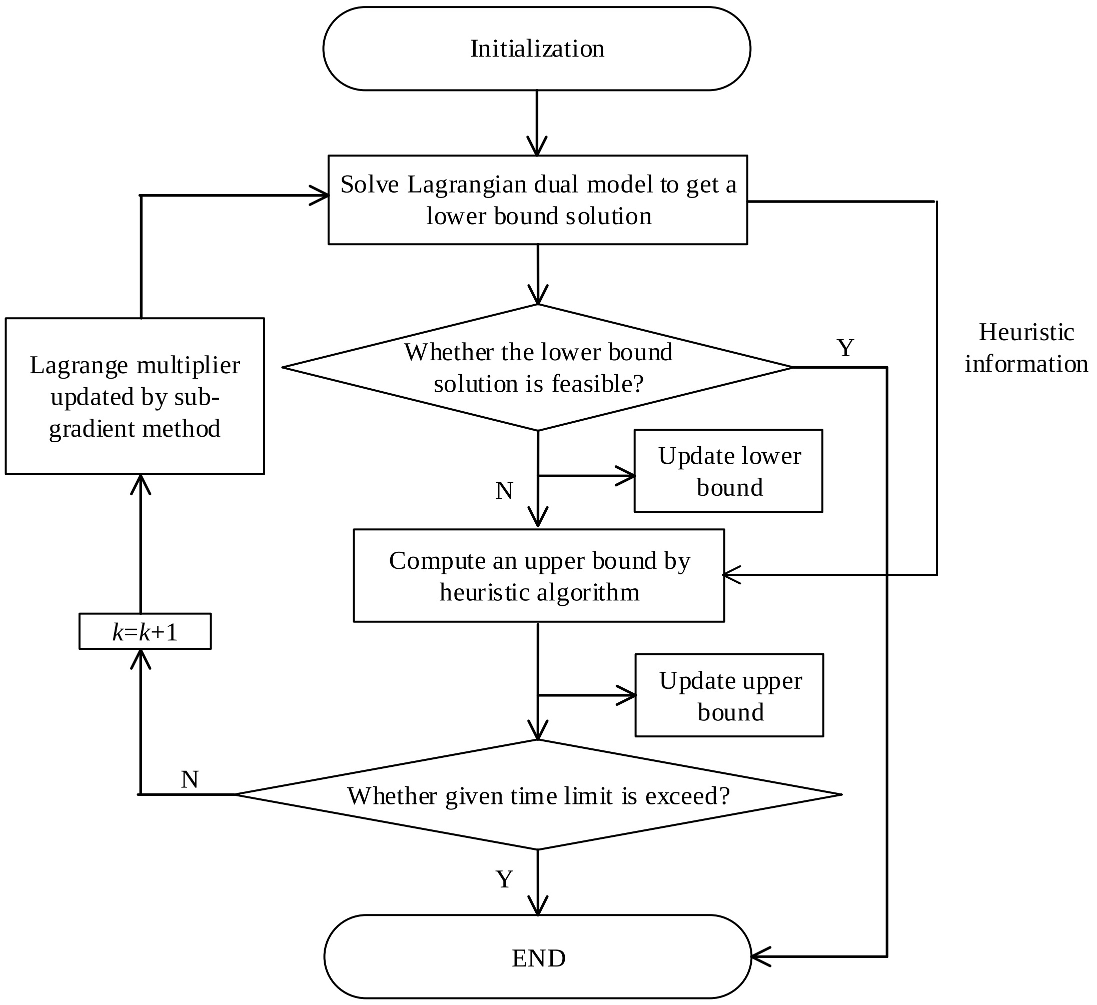

Initialization
Solve Lagrangian dual model to get a lower bound solution
Heuristic information
Lagrange multiplier updated by sub-gradient method
Whether the lower bound solution is feasible?
Y
N
Update lower bound
Compute an upper bound by heuristic algorithm
k=k+1
Update upper bound
N
Whether given time limit is exceed?
Y
END

Fig. 4 Flow chart of Lagrangian heuristic

6.2 Solving the Lagrangian dual model

Since the TSSN is a topologically ordered sparse graph, we use the Forward Dynamic Programming (FDP) algorithm to solve the shortest path problem. The algorithm to solve M4 consists of the following three steps.

Step1: Initialize the crew member set P and the pairing set D, and set $p = 1$. For $\forall l \in L$, $\forall(i,j) \in A_{tss}^{l}$ and $\varphi_i = \varphi_j = l$, set tt_{ij}=$tt_{ij} - \rho^k(l)$.

Step2: Use FDP algorithm to find the shortest path d_p, which is a pairing carried out by crew member p. For $\forall(i,j) \in A_{tss}^{d}$ and $x_{ij}^{p} = 1$, set $tt_{ij} = M$ (M is a large positive number), and add d_p into D.

Step3: If $p = |P|$, the algorithm ends. Otherwise, set $p = p + 1$, then go to step 2.

In Step2, we realize Constraints (6) to (9) by setting the arc lengths, which have been covered by newly generated pairings, to avoid that these are covered in the next iteration.

6.3 Updating the Lagrange multiplier

The objective value of M4 changes with Lagrange multiplier ρ. Therefore, in the iterative process, the value of ρ needs to be constantly updated by using the subgradient method shown as formula (17).

$$\rho^{k+1}(l) = \max\left\{0, \rho^k(l) + \sigma^k\left(1 - \sum_{p\in P}\sum_{(i,j)\in A_{tss}^l, \varphi_i=\varphi_j=l} x_{ij}^p\right)\right\} \tag{17}$$

σ^k indicates the iteration step length in the kth iteration. This is defined as necessary, but this needs to meet the following rules.

$$\sum_{k=1}^{\infty} \sigma^k = \infty \text{, and } k \to \infty, \sigma^k \to 0$$

6.4 Obtaining an upper bound

We design the following algorithm to get an upper bound for the improved model.

Step1: Initialize the crew member set P and the pairing set D, and set $p = 1$. Define the number of times that task l is covered in the lower bound solution as nLB_l. For $\forall l \in L$, set $nLB_l = 0$. Similarly, define the number of times that task l is covered in the upper bound solution as nUB_l. For $\forall l \in L$, set $nUB_l = 0$. Define M as a large positive number.

Step2: For $\forall l \in L$, calculate nLB_l according to the lower bound solution obtained in this iteration. For $\forall (i,j) \in A_{tss}^l$ and $\varphi_i = \varphi_j = l$, set $tt_{ij} = tt_{ij} - \rho^k(l)$.

Step3: Use the FDP algorithm to find the shortest path d_p .

Step4: If d_p only covers the virtual arc between the depots and the length of the virtual arc tt_{od} is less than M, go to Step 5. Otherwise, add d_p into D , and for $\forall (i,j) \in A_{tss}^l$ and (i,j) covered by path d_p , set $tt_{ij} = M$, and go to Step 6.

Step5: For $\forall l \in L$, calculate nUB_l. If $nUB_l = 0$, set the length of the virtual arc $tt_{od} = (n' + 1) * M$, where n' is the maximum number of tasks in a feasible duty. Then use the FDP algorithm to find the shortest path d_p , set $tt_{od} = -M$, and go to Step 4; otherwise, go to Step 6.

Step6: If $p = |P|$, the algorithm ends. Otherwise, set $p = p + 1$, and go to Step 3.

7 Computational results

We apply the proposed approach to solve the railway CSP for managing an intercity high-speed rail line and a regional high-speed rail network. We use two strategies to decrease the size of TSSN when solving these cases.

- *Concrete consecutive tasks*, which are operated by the same train units when the dwell time between consecutive tasks is too short to allow a shift, into one task.
- *Set limits on minimum working time* Td_{min}, the maximum transfer time Tt^{max}, the minimum consecutive driving time To_{min}, the maximum break time Tr^{max} and the maximum overnight rest time Ts^{max}, according to practical experience.

We terminate the Lagrangian heuristic when the iteration times exceed 100 units. The approach is coded in C#. The computation is performed on an Intel(R) Core(TM) i7-7500U 2.70GHz, 8.00GB RAM personal computer.

7.1 Case 1: An intercity high-speed rail line

We first test our approach on the railway CSP of an intercity high-speed rail line, BJ-TJ intercity high-speed Rail Line (cited as JJRL in this paper). In this case, 158 trains run between the origin station and the destination station of JJRL. The average running time of a train is 30 minutes. Because the running time of a train is short and nearly 75% of the trains do not stop at any intermediate station, we directly use a train as a task in this case. There are two depots, BJN and TJ, located at the origin station and the destination station of JJRL, respectively. The values of the parameters are reported in Table 3.

Table 3 Values of the parameters in Case 1

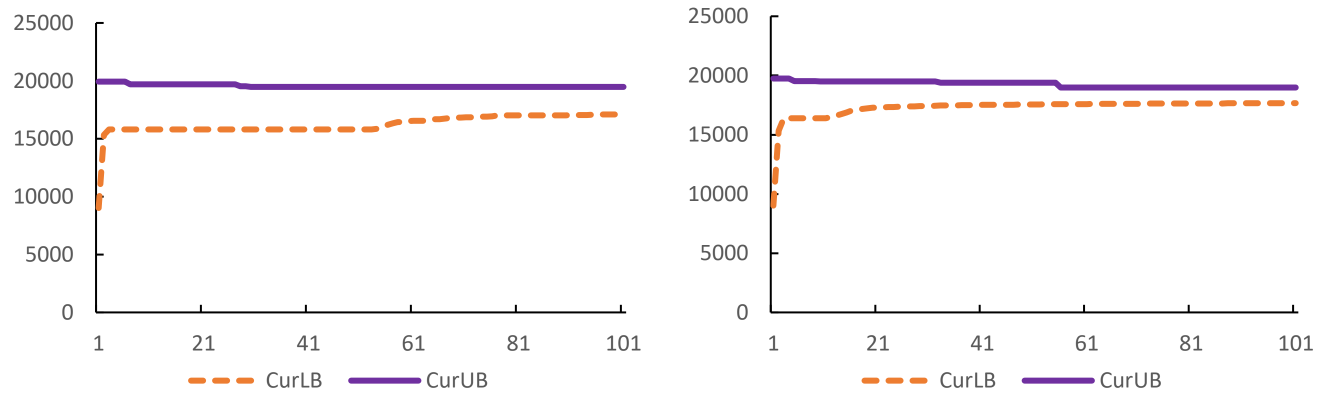
Parameters	Value
Number of tasks (trains)	158
Number of stations	2
Number of crew members in each Depot	30
Total number of crew members	60
Iteration step	$\sigma^k = 40/(k+1)$

7.1.1 Effect of inequality constraints on strengthening the formulation

Firstly, we construct the TSSN for Case 1. Since the number of the decision variables is up to 393,600, it is difficult to directly solve M1 by using CPLEX, because of the limited PC memory. We compare the difference in solving M2 and M3 to test the effects of strengthening the formulation. We set the lunch time window to [11:00,13:00] and the supper time window to [17:00,19:00]. The computational results and the convergence curves are shown in Table 4 and Fig. 5, respectively.

Table 4 Computation results of solving M2 and M3 in Case 1

Indicators	Value	
	M2	M3
Construction time of TSSN	0.70s	
Number of decision variables	393600	
Upper bound solution	19477.55	18986.01
Lower bound solution	17096.84	17667.00
Optimality Gap	12.22%	6.95%
Solution time	723.36s	816.91s
Iteration in which the optimal solution first appears	28	56
Number of crew members needed in the BJN depot	25	24
Number of crew members needed in the TJ depot	25	23
Number of times of crew having lunch	0	3
Number of times of crew having supper	1	2



a. Convergence curves of solving M2 b. Convergence curves of solving M3

Fig. 5 Convergence curves of solving M2 and M3 in Case 1.

From the above results, we can find that the gap between the lower bound (LB) and upper bound (UB) of solving M2 is larger than that when solving M3. The solution obtained from solving M3 is also better than the solution obtained from solving M2. However, solving M3 may take a little more time than solving M2. We will next discuss the performance of strengthening the formulation of CSP with different network structures (in Section 7.2.1).

7.1.2 Sensitivity analysis of the length of meal time window

We analyze the effect of different lengths of meal time windows for four cases. We set 1 hour, 2 hours, 3 hours and 4 hours as the lengths of meal time windows in Case1-1, Case 1-2, Case 1-3 and Case 1-4. There is no feasible solution to Case 1-1. The computational results of the other three cases are shown in Table 5.

Table 5 Computation results with different lengths of meal time windows in Case 1

Indicators	Value		
	Case 1-2	Case 1-3	Case 1-4
Length of meal time window	2h	3h	4h
Lunch time window	[11:00,13:00]	[10:30,13:30]	[10:00,14:00]
Supper time window	[17:00,19:00]	[16:30,19:30]	[16:00,20:00]
Construction time of TSSN	0.70s	0.77s	0.84s
Number of decision variables	393600	501300	559200
Upper bound solution	18986.01	19028.02	18988.01
Lower bound solution	17667.00	17555.67	17495.54
Optimality Gap	6.95%	7.74%	7.86%
Solution time	816.91s	926.88s	1123.11s
Number of crew members needed in the BJN depot	24	23	24
Number of crew members needed in the TJ depot	23	24	23
Total number of crew members needed	47	47	47
Number of times of crew having lunch	3	2	1
Number of times of crew having supper	3	2	0

We can draw the following conclusions:

- As the length of meal time window increases, the size of TSSN, the number of decision variables and the solution time increase. Although the construction of TSSN in these four cases are very fast (less than one second), the number of decision variables is very large (up to 559200).
- When the length of the meal time window is too small, there may be no feasible solution, i.e. in Case 1-1.
- The length of the meal time window does not affect the total number of crew members needed, which are the same in Case 1-2, Case 1-3 and Case 1-4.
- The length of the meal time window may affect the times that the crew members have the meal during the duty. For longer meal time windows, the start of the meal time window is earlier, then there may be more duty starting with meal status of *after lunch/supper*.

7.2 Case 2: a regional high-speed rail network

We now test our approach on the railway CSP of a regional high-speed rail network. The structure of this rail network is illustrated in Fig. 6. There are four depots: NJ, SH, HZ, and HF. The values of the parameters are mentioned in Table 6.

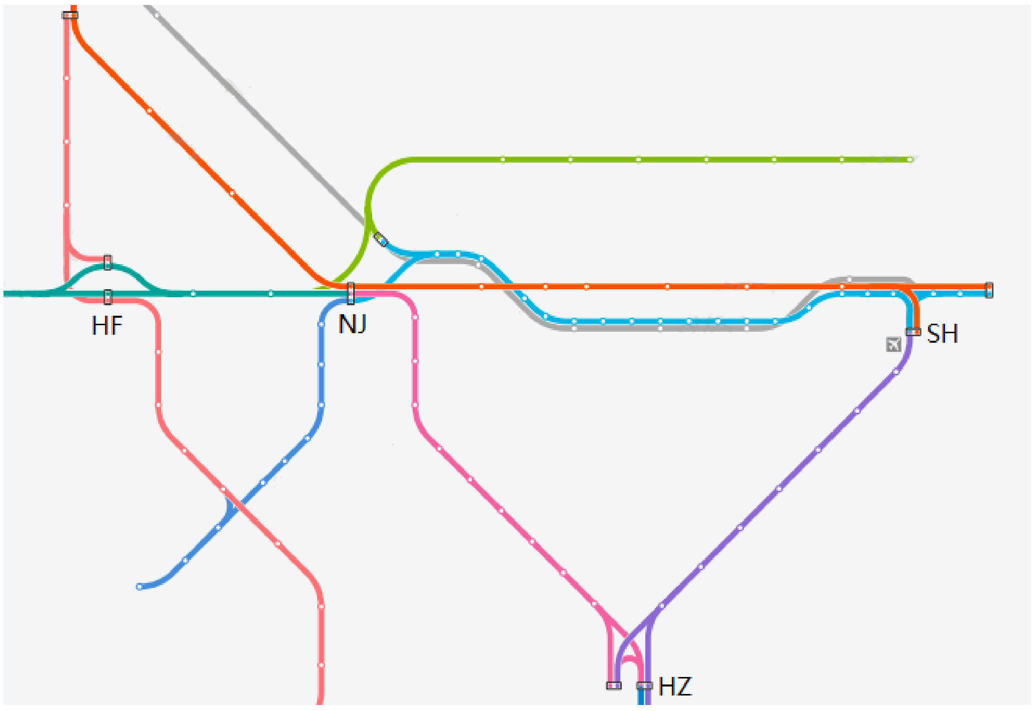


Fig. 6 Regional high-speed rail network structure

Table 6 Values of the parameters in Case 2

Parameters	Value
Number of tasks	2107
Number of trains	345
Number of crew members in the NJ depot	80
Number of crew members in the SH depot	80
Number of crew members in the HZ depot	40
Number of crew members in the HF depot	40
Total number of crew members	240
Iteration step	$\sigma^k = 400/(k+1)$

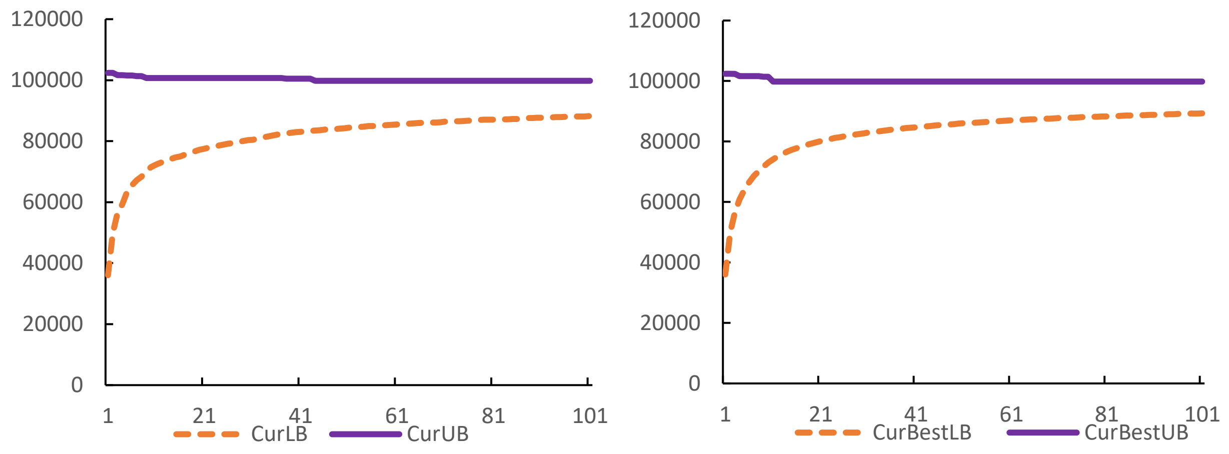
7.2.1 Effect of inequality constraints on strengthening the formulation

We set the lunch time window to [11:00,13:00] and the supper time window to [17:00,19:00] to compare the performance of strengthening the formulation in Case 2. The computational results and the convergence curves are shown in Table 7 and Fig. 7, respectively.

Table 7 Computation results of solving M2 and M3 in Case 2

Indicators	Value	
	M2	M3
Construction time of TSSN	1.51s	
Number of decision variables	2568480	
Upper bound solution	99791.82	99814.82
Lower bound solution	88250.20	89319.58
Optimality Gap	11.57%	10.51%
Solution time	8921.46s	8863.68s
Iteration when the optimal solution first appears	44	11
Number of crew members needed in the NJ depot	55	57

Number of crew members needed in the SH depot	58	58
Number of crew members needed in the HZ depot	18	15
Number of crew members needed in the HF depot	23	23
Total number of crews needed	154	153
Number of times of crew having lunch	1	2
Number of times of crew having supper	4	4



a. Convergence curves of solving M2 b. Convergence curves of solving M3

Fig. 7 Convergence curves of solving M2 and M3 in Case 2.

In Case2, the solution obtained from solving M3 is also better than the solution obtained from solving M2. However, comparing the results of Fig. 7 with the ones of Fig. 5, the difference between the convergence curve related to M2 and M3 is not as obvious as in Case 1. The different performance of strengthening the formulation is mainly caused by the difference in the train schedule structure. In Case 1, most of the trains have the same running time. Thus, most of the duties covering different trains, but with the same running time may have the same working time. In this situation, many equivalent duties will lead to slow convergence. By limiting duties with the same content, the gap between LB and UB decreases quicker. In Case 2, a higher diversity of trains increases the diversity of duties but decreases the equivalency of duties. Thus, the difference of convergence curves of solving M2 and M3 maybe not as obvious as that in Case 1.

7.2.2 Sensitivity analysis of the length of meal time windows

In Case 2, we also set 1 hour, 2 hours, 3 hours and 4 hours as the lengths of meal time windows in Case2-1, Case 2-2, Case 2-3 and Case 2-4. There is still no feasible solution in Case 2-1. The computational results of the other three cases are shown in Table 8.

Table 8 Computation results with different lengths of meal time windows in Case 2

Indicators	Value		
	Case 2-2	Case 2-3	Case 2-4
Length of meal time window	2h	3h	4h
Lunch time window	[11:00,13:00]	[10:30,13:30]	[10:00,14:00]
Supper time window	[17:00,19:00]	[16:30,19:30]	[16:00,20:00]
Construction time of TSSN	1.51s	1.94s	2.39s
Number of decision variables	2568480	3182400	3443280

Upper bound solution	99814.82	99891.78	99284.15
Lower bound solution	89319.58	89101.63	88992.65
Optimality Gap	10.51%	10.80%	10.37%
Solution time	8863.68s	10899.26s	13110.31s
Number of crew members needed in the NJ depot	57	55	50
Number of crew members needed in the SH depot	58	53	52
Number of crew members needed in the HZ depot	15	19	20
Number of crew members needed in the HF depot	23	23	24
Total number of crews needed	153	150	146
Number of times of crew having lunch	2	3	2
Number of times of crew having supper	4	5	3

The computation results in Case 2-1 to Case 2-4 reflect similar phenomenon with that in Case 1-1 to Case 1-4. However, as the length of meal time window increases, the total number of needed crew decreases.

In Case 1, the frequency of trains is high and all trains have the same origin and destination. Therefore, this is probably due to a more convenient arrangement of the meal break in the short time window, without the need of increasing the number of needed crew members. In Case 2, the frequency of trains at different stations changes a lot. For example, 248 trains originate from one of the four depots, while 97 trains originate from one of the other 23 stations. Therefore, for a station with the low frequency of trains, when the length of meal time window is short, the crew usually have more break time to have lunch/supper, because of less feasible connection tasks. Thus, more crew members would be needed in order to create a better crew plan.

8 Conclusions

This paper proposed a time-space-state network to describe the 'mixed time constraints' of the railway CSP. Based on the TSSN, the "feasible duty generation" could be transformed into a classic shortest path problem. Therefore, we constructed an initial network flow model based on the TSSN. To break the symmetry of the initial model and to strengthen the formulation, we added five sets of inequalities into the initial model and generated an improved model. The improved model was then solved by a Lagrangian heuristic. Finally, we evaluated the approach on real-world instances under two typical rail network structures and got the following conclusions:

- By solving the improved model, we could get better solutions than when solving the initial model. And the efficiency of solving the improved model was also better than that of solving the initial model.
- Different train schedule structure caused different performance of inequality constraints to strengthen the formulation on the railway CSP.
- As the length of the meal time window increased, the TSSN size, the number of decision variables and the solution time also increased.
- The length of the meal time window affected the number of crew members needed and the number of times that the crew members had meals during the duty. For longer meal time windows, the fewer crew members were needed.

To further improve the proposed solution method, it would be interesting to embed a column generation method into the developed Lagrangian framework.

Acknowledgments

This work was supported by National Key Research & Development Plan of China (Grant No. 2018YFB1201403).

References

Abbink, E. (2014). Crew management in passenger rail transport. Ph.D thesis, Erasmus University Rotterdam, Rotterdam.

Abbink, E., Albino L, Dollevoet T, Huisman, D., Roussado, J., & Saldanha, R.L. (2011). Solving Large Scale Crew Scheduling Problems in Practice. *Public Transport*, *3*(2),149-164.

Abbink, E., Huisman, D., & Kroon, L. (2018). Railway Crew Management. Handbook of Optimization in the Railway Industry, Springer International Publishing, 243-264.

Arabeyre, J., Fearnley, J., Steiger, F., & Teather, W. (1969). The airline crew scheduling problem: A survey. *Transportation Science*, *3*, 140–163.

Barnhart, C., Belobaba, P., & Odoni, A.R.(2003). Applications of operations research in the air transport industry. *Transportation Science*, *37*(4), 368-391.

Caprara, A., Fischetti, M., Guida, P. L., Toth, P., & Vigo, D. (1999). Solution of large-scale railway crew planning problems: the Italian experience. *Computer-Aided Transit Scheduling*, *471*, 1-18.

Caprara, A., Fischetti, M., Toth, P., Vigo, D., & Guida, P. L. (1997). Algorithms for Railway Crew Management. *Mathematical Programming*, *79* (1-3), 125-141.

Caprara, A., Kroon, L., Monaci, M., Peeters, M., & Toth, P. (2007). Passenger railway optimization. Handbooks in operations research and management science, transportation, Elsevier B.V., Amsterdam, 129–187.

Chen, S., Shen, Y., Su, X., & Chen, H. (2013). A Crew Scheduling with Chinese Meal Break Rules. *Journal of Transportation Systems Engineering and Information Technology*, *13*(2), 90-95.

Desrochers, M., & Soumis, F. (1989). A Column Generation Approach to the Urban Transit Scheduling Problem. *Transportation Science*, *23*(1), 1-14.

Ernst, A. T., Jiang, H., Krishnamoorthy, M., & Sier, D. (2004). Staff scheduling and rostering: a review of applications, methods and models. *European Journal of Operational Research*, *153*(1), 3-27.

Freling, R., Lentink, R.M., & Odijk, M.A. (2000). Scheduling Train Crews: A Case Study for the Dutch Railways. *Econometric Institute Report*, *505*, 153-166.

Gopalakrishnan, B., & Johnson, E.L. (2005). Airline Crew Scheduling: State-of-the-Art. *Annals of Operations Research*, *140*(1), 305-337.

Hoffmann, K., & Buscher, U. (2018). Valid inequalities for the arc flow formulation of the railway crew scheduling problem with attendance rates. *Computers & Industrial Engineering*. Available online.

Huisman, D. (2007). A column generation approach for the rail crew rescheduling problem. *European Journal of Operational Research*, *180*(1)：163-173.

Huisman, D., Freling, R., & Wagelmans, A. P. M. (2005). Multiple-Depot Integrated Vehicle and Crew Scheduling.

Transportation Science, *39*(4), 491-502.

Jütte, S., & Thonemann, U.W. (2015). A Graph Partitioning Strategy for Solving Large-scale Crew Scheduling Problems. *OR Spectrum*, *37*(1), 137-170.

Kasirzadeh, A., Saddoune, M., & Soumis, F. (2017). Airline crew scheduling: models, algorithms, and data sets. *Euro Journal on Transportation & Logistics*, *6*(2), 111-137.

Kohl, N. (2003). Solving the world's largest crew scheduling problem. ORt 8–12.

Lourenço, H.R., Paixão, J., & Portugal, R. (2001). Multi-objective Metaheuristics for the Bus Driver Scheduling Problem. *Transportation Science*, *35*(3), 331-343.

Mahmoudi, M., & Zhou, X. (2016). Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state–space–time network representations. *Transportation Research Part B: Methodological*, *89*, 19-42.

Nishi, T., Muroi, Y., & Inuiguchi, M. (2011). Column Generation with Dual Inequalities for Railway Crew Scheduling Problems. *Public Transportation*, *3*(1), 25-42.

Sahin, G., & Yüceoğlu, B. (2011). Tactical Crew Planning in Railways. *Transportation Research Part E: Logistics and Transportation Review*, *47*(6), 1221-1243.

Sherali, H. D., Bish, E. K., & Zhu, X. (2006). Airline fleet assignment concepts, models, and algorithms. *European Journal of Operational Research*, *172*(1), 1-30.

Tallys, H. Y., Arnaldo, V. M., & Cid, C. S. (2005). Hybrid column generation approaches for urban transit crew management problems. *Transportation Science, 39*(2), 273-288.

Vaidyanathan, B., Jha, K.C., & Ahuja, R.K. (2007). Multi-commodity Network Flow Approach to The Railroad Crew-scheduling Problem. *IBM Journal of Research and Development, 51*(3.4), 325-344.

Wang, Y., Liu, J., Miao, J. (2009). Modeling and Solving the Crew Scheduling Problem of Passenger Dedicated Line. *Journal of The China Railway Society(in Chinese)*, *31*(1),15-19.

Appendix

Appendix A: Notations

Sets:	
G_{ts}	Basic time-space network
V_{ts}	Set of all vertices in the network G_{ts}
A_{ts}	Set of all arcs in network G_{ts}.
V_{ts}^{o}	Set of virtual source vertices
V_{ts}^{l}	Set of tasks' arrival vertices and departure vertices
V_{ts}^{d}	Set of virtual sink vertices
A_{ts}^{o}	Set of on-duty arcs
A_{ts}^{l}	Set of task arcs

A_{ts}^{g}	Set of ground arcs
A_{ts}^{on}	Set of overnight arcs
A_{ts}^{d}	Set of off-duty arcs
A_{ts}^{od}	Set of virtual arcs
G_{tss}	time-space-state network
V_{tss}	Set of all vertices in the network G_{tss}
A_{tss}	Set of all arcs in the network G_{tss}
L	Set of tasks
P	Set of available crew members
D	Set of pairings
Ω_i	set of all states of vertex v_i
$\omega_{i(m)}$	set of attributes of the state of $v_{i(m)}$

Decision variables:

x_{ij}^{p}	If arc (i, j) covered by crew member p, x_{ij}^{p} =1; otherwise, x_{ij}^{p} =0
e_{ij}^{pq}	If the crew pairing p and q do not cover task arc (i, j) at the same time, e_{ij}^{pq} = 1; otherwise, e_{ij}^{pq} = 0

Parameters:

Td	Working time of a duty
Td^{max}	Maximum working time of a duty
Tt	Transfer time
Tt_{min}	Minimum transfer time
To	Consecutive driving time of a duty
To^{max}	Maximum consecutive driving time of a duty
Tr	Break time
Tr_{min}	Minimum break time
Ts	Overnight rest time
Ts_{min}	Minimum overnight rest time
Tp	Period of a pairing
Dd	The maximum period of a pairing
TW_{MB}^{l}	Break time for lunch
TW_{MB}^{s}	Break time for supper
ML_{min}	Lower bound of lunch time window
ML_{max}	Upper bound of lunch time window
MS_{min}	Lower bound of supper time window
MS_{max}	Upper bound of supper time window
Te	Dining time
Te_{min}	Minimum meal time

Te_b^a	The time that meal break begins after the start of the duty
Te_f^b	The time that meal break finishes before the end of the duty
$v_{i(m)}$	The time-space-state vertex
$Td_{i(m)}$	Accumulated working time of vertex $v_{i(m)}$ in a duty
$tt_{j(n),i(m)}$	The length of the arc ($v_{j(n)}$, $v_{i(m)}$)
$To_{i(m)}$	Accumulated consecutive driving time of vertex $v_{i(m)}$ in a duty
$Tc_{i(m)}$	Accumulated break/waiting time of vertex $v_{i(m)}$ in a duty
$Tp_{i(m)}$	Accumulated working time of vertex $v_{i(m)}$ in a pairing
$Mb_{i(m)}$	Dining status for lunch and supper
φ_i	index of task
t_i	arrival/departure time of the task φ_i
s_i	arrival/departure station of task φ_i
θ_i	index of train units routing covering task φ_i
$t_{i(m)}$	departure/arrival time of vertex $v_{i(m)}$
$s_{i(m)}$	departure/arrival station of vertex $v_{i(m)}$
$\rho(l)$	Lagrange multiplier of task l
M	A large positive integer

Appendix B: Algorithm generating TSSN

```
// Algorithm GenerateTSSN()
Begin
  $V_{tss} = V_{ts}$;
  $V_c = \emptyset$;
  for each depot vertex $o \in V_{ts}^o$ do
      $V_c = V_c \cup \{o\}$;
      $\omega_o = \{0,0,0,0,0\}$;
      while $V_c \neq \emptyset$ do
          for each $i \in V_c$ do
              $V_c = V_c - \{i\}$;
              for each $j \in V_{ts}$ and $\exists(i,j) \in A_{ts}$ do
                  $V_c = V_c \cup \{j\}$;
                  for each $\omega_{i(m)} \in \Omega_i$ do
                      $n = |\Omega_j|$;
                      CaculateState($\omega_{j(n+1)}, \omega_{i(m)}$);
                      if FeasibleState($\omega_{j(n+1)}$) returns TRUE
                          if CheckSameState($\omega_{j(n+1)}$) returns NULL
                              Create new vertex $j(n+1)$ with attribute($t_j, s_j, \varphi_j, \theta_j, \omega_{j(n+1)}$);
                              $V_{tss} = V_{tss} \cup \{j(n+1)\}$;
                              Create new arc($i(m), j(n+1)$) with attribute($tt_{ij}, \theta_{ij}$);
                              $A_{tss} = A_{tss} \cup \{(i(m), j(n+1))\}$;
                          else returns $\omega_{j(k)}$
                              Create new arc($i(m), j(k)$) with attribute($tt_{ij}, \theta_{ij}$);
```

```
                        $A_{tss} = A_{tss} \cup \{(i(m), j(k))\}$;
                    end;
                end;
            end;
        end;
    end;
  end;
end;
```

In the algorithm, the function of CaculateState($\omega_{j(n+1)}, \omega_{i(m)}$) is calculating state $\omega_{j(n+1)}$of vertex v_j according the state $\omega_{i(m)}$ of vertex v_i using the method introduced in Section4.1 The function of FeasibleState($\omega_{j(n+1)}$) is to verify whether the state $\omega_{j(n+1)}$is feasible according to the crew rules introduced in Section3.2. The function of CheckSameState ($\omega_{j(n+1)}$) is to check whether there is the same state of vertex v_j as $\omega_{j(n+1)}$. If there is no the same state, return NULL; Otherwise, return the state node represented by $\omega_{j(k)}$.