One-block train formation in large-scale railway networks: An exact model and a tree-based decomposition algorithm

Chongshuang Chen[1,2], Twan Dollevoet[*,3,4], Jun Zhao[5]

[1] School of Mathematics, Southwest Jiaotong University, Chengdu, Sichuan 610031, China

[2] Rotterdam School of Management, Erasmus University Rotterdam, Rotterdam, The Netherlands

[3] Econometric Institute, Erasmus University Rotterdam

P.O. Box 1738, 3000 DR Rotterdam, The Netherlands, dollevoet@ese.eur.nl

[4] Erasmus Center for Optimization in Public Transport, Rotterdam, The Netherlands

[5] School of Transportation and Logistics, Southwest Jiaotong University, Chengdu, Sichuan 610031, China

Econometric Institute Report Series – EI-2017-32

Abstract

We investigate the one-block train formation problem (TFP) in the railway freight transportation industry given a car route for each shipment. The TFP considers both the block design and the car-to-block assignment in the tactical level. Moving beyond current researches on service network design, the unitary rule and the intree rule are taken into account in this study based on the Chinese railway background. We develop a linear binary programming formulation to minimize the sum of train cost and classification delay subject to limitations on the classification capacity and the number of sort tracks at each station. Furthermore, we propose a novel solution methodology that applies a tree-based decomposition algorithm. Here, we first decompose the whole network into a series of rooted trees for each destination separately. Then, we divide the trees into sufficiently small subtrees, whose size is regulated by a node size parameter. Finally, we construct a restricted linear binary model for each subtree and solve these models sequentially to find their optimal solutions. Our computational results on a realistic network from the Chinese railway system with 83 stations, 158 links and 5700 randomly generated demands show that the proposed algorithm can derive high-quality solutions within 3 hours. These solutions are on average 43.89% better than those obtained after solving the linear binary program for 1 day.

Keywords: Railway Freight Transportation; Train Formation Problem; Service Network Design; Tree-based Decomposition; Arborescence Structure

1 Introduction

In modern society, transportation is of essential importance for global logistics supply chains and strongly correlates with world economic growth. Railway transportation has incomparable advan-

*Corresponding author

tages over other modes of transportation with its 24/7/365 service and large capacity for line-haul transportation. In China, the national railroad accomplished a 40% share in passenger transportation and a 13% share in freight transportation in 2015 (National Bureau of Statistics of China 2015).

The rail network consists of a large number of *terminals* and *tracks*. Among the terminals, *marshaling yards*, also called *shunting yards* or *classification yards*, are equipped with a hump and several shunting locomotives. The *demand* for transportation is specified as a set of requests to transport cars between a given origin and destination. Movements of cars on the network are performed by so-called train services, which represent a train being operated from one station to another. The marshaling yards play a central role in the way cars and trains are handled in the railway system. At these stations, cars can be transferred from one train service to another.

For significantly important goods, or regular and high-volume demands, rail carriers provide a direct train from the freight's origin to its destination. Such a direct train is referred to as a complete train in Italy (Lulli et al. 2011), a dedicated train in North America (Zhu et al. 2014), or an entire train in China (Lin et al. 2012). In contrast, to reduce the handling cost and delay time, most demands share trains with others from different origins and/or to different destinations (Guastaroba et al. 2016). For those demands, it must be determined how to aggregate all cars into trains through the network in the tactical level. This process is named the *train formation problem* (TFP) in China. The results of the TFP guide the train scheduling process in the operational level.

To benefit from economies of scale, railroads generally solve the TFP in two phases: firstly cars are grouped into so-called *blocks*; then blocks are combined to make up trains. Cars in the same block may pass through a series of intermediate yards without being disassembled before they reach the destination of the block. This process is known as *consolidation*, and it is widely applied in the freight transportation sector, e.g., for postal and package delivery, by trucking carriers, in container shipping, and in the airline industry (Ahuja et al. 2007). In a railway system, consolidation is realized by a series of operations in marshaling yards. First, each car is assigned to a sort track that is designated for a specific block. Second, some blocks are bound to form an outbound train if the number of cars for that train accumulates to a certain quantity. These sorting and binding operations on cars and blocks are generally referred to as *classification*. It should be noted that entire blocks may be switched from one train to another without classification, which is called a *block swap*. For more details, please see the survey of Boysen et al. (2012).

In Chinese practice, the TFP has some specific characteristics which are ignored in most of the literature. Nowadays, almost all trains carry one block, and each block is delivered to its destination via no more than one train. This kind of train is vividly called a *single group train* or a *one-block train*, in contrast to a *multi-block train* carrying several blocks. In China, neither multiple paths nor diverse classification policies are allowed for a demand. Thus, all cars in a demand follow the same path and the same classification policy. We refer to this requirement as the *unitary rule* in this paper. Another compulsive regulation that arises in Chinese railroads is named the *intree rule* in this paper. This rule states that all cars with the same destination must be operated in exactly the same way once they meet at an intermediate yard. There are two potential benefits of the intree rule. First of all, for any shipment, regardless of its origin, given any present classification station and final destination, its next classification location is specified uniquely. This kind of memoryless policy

is not only easy when making decisions, but also convenient to implement in practice. Secondly, in combination with the unitary rule, the intree requirement ensures predictability and tractability on how each car gets handled throughout the network.

As far as we know, both the unitary rule and the intree rule are rarely encountered in literature. Almost all existing works on service network design assume that demands can be split and may use more than one path (Crainic 2000, Wieberneit 2008). However, such an assumption may not cover applications in transportation systems where the unitary rule holds, for example, express package delivery (Barnhart et al. 2002) or hazardous material transportation (Verter and Kara 2008). The intree rule is only applied by Bodin et al. (1980), Lin et al. (2012) and Fugenschuh et al. (2015) in railways, and by Jarrah et al. (2009) in less-than-truckload transportation, where it is called a pure strategy constraint, a reclassification strategy, a unique successor rule or a load-planning requirement, respectively.

There has been limited success applying models based on North American or European rail networks to the Chinese context. The main difference is that accumulation cost caused by staying on classification tracks are more important than frequency cost in China, as stated in Lin et al. (2012). In this paper, we study the TFP and incorporate the unitary rule and the intree rule. Our setting is the same as that in Lin et al. (2012). However, both the mathematical formulation and the solution approach are different. We build a linear binary program by utilizing path-based variables and well-designed linear constraints, whereas a non-linear, recursive model with arc-based variables is developed by Lin et al. (2012). Our exact model can be used to find provably optimal solutions for small and medium-scale networks. Additionally, we propose a novel tree-based decomposition strategy that is guaranteed to find a feasible solution, if one exists. Lin et al. (2012) relax the classification capacity and sort track constraints and design a simulated annealing algorithm. To conclude, our methodology provides a guaranteed upper bound within acceptable computation time for large-scale instances.

Moving beyond current researches on service network design, our contributions are fourfold:

- We propose a linear binary programming formulation for the one-block train formation problem incorporating the unitary rule and the intree rule based on the Chinese railway background, see §4.2.
- We develop a tree-based decomposition algorithm dividing the whole network into manageable subtrees subject to any predefined node size parameter, see §5.1.
- For each subtree, we propose an active-lazy strategy on variables and reserve-release-adjust manipulations on resource utilization to construct a restricted model, see §5.2.
- We demonstrate the applicability of our methods to large-scale instances based on a real-world railway system, see §6.

The remainder of this paper is organized as follows. We describe the train formation problem in §2. In §3, we review the literature on train formation in railway freight transportation. §4 introduces the mathematical formulation for the one-block TFP. A tree-based decomposition algorithm and some

acceleration techniques are proposed in §5. §6 discusses computational results, and §7 summarizes our conclusions.

2 Problem description

We begin this section with a brief description of the train formation problem. Then, we discuss the tree structure that is imposed on solutions by the intree rule.

As introduced in §1, the consolidation strategy in railway freight transportation contains two phases. First, cars are consolidated into blocks, and then blocks are consolidated into trains. Specifically, the task of the TFP could be subdivided as follows:

1. Which blocks (pairs of yards) are built;
2. Which cars are grouped to which blocks;
3. Which trains (pairs of yards) are provided;
4. Which train takes which blocks.

The above four subproblems have other common terminologies in literature as well. The first subproblem is referred to as block design (Ireland et al. 2004), railroad blocking (Newton et al. 1998, Barnhart et al. 2000, Ahuja et al. 2007), or the block policy (Cordeau et al. 1998). The second subproblem is referred to as car-to-block assignment (Kwon et al. 1998) or traffic distribution (Cordeau et al. 1998, Zhu et al. 2014). The fourth subproblem is known as block-to-train assignment (Jha et al. 2008), making-up (Khaled et al. 2015), or makeup policy (Cordeau et al. 1998).

We present a simple example in Figure 1 to explain the above subproblems. In the figure, we see 4 stations on a line and 6 demands. Recall that the path for each demand is given as input to the TFP. In this case, all demand are transported via the shortest path. In the first phase, the 6 demands are consolidated into 4 blocks. The car-to-block assignment (CBA) is detailed in Table 1. In the second phase, these 4 blocks are consolidated into 1 multi-block train that carries 3 blocks and 1 one-block train. The block-to-train assignment (BTA) is listed in Table 2.

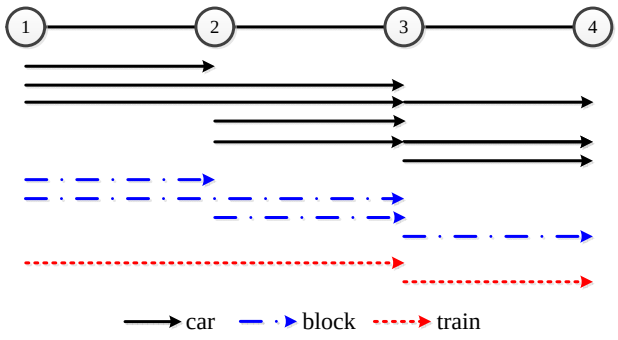


Figure 1: A simple example for the TFP

In the TFP, the route for each demand is given as input. As a consequence, the path for a block and a train are inherently determined by the car route. Furthermore, the frequency of the

Table 1: Car-to-block assignment

No.	Demand	Assigned block(s)
1	$1 \to 2$	$1 \to 2$
2	$1 \to 3$	$1 \to 3$
3	$1 \to 4$	$1 \to 3, 3 \to 4$
4	$2 \to 3$	$2 \to 3$
5	$2 \to 4$	$2 \to 3, 3 \to 4$
6	$3 \to 4$	$3 \to 4$

Table 2: Block-to-train assignment

No.	Block	Assigned train(s)	Train type
1	$1 \to 2$	$1 \to 3$	multi-block
2	$1 \to 3$	$1 \to 3$	multi-block
3	$2 \to 3$	$1 \to 3$	multi-block
4	$3 \to 4$	$3 \to 4$	one-block

train is indirectly determined by the cars contained in the blocks that are carried by the train. For the one-block TFP that we consider, there is a one-to-one correspondence between blocks and trains. As a result, the third subproblem is completely equivalent to the first one and the BTA is not necessary. Consequently, the one-block TFP reduces to the combination of block design and car-to-block assignment.

The TFP has two types of costs. The first type is for providing train services and is measured as the accumulation cost in marshaling yards to aggregate a sufficient number of cars to make up a train. For example, 50 cars are required for a train in China (Lin et al. 2012). Another type is delay cost for demands due to classification on intermediate yard(s). All costs are measured in car*hour and reflect the practical situation where full advantage of mobile equipment is necessary, given a fixed equipment capacity shortage in China. The TFP is mainly constrained by the classification capacity and the number of sort tracks in yards. The capacity of tracks between stations are already taken into account when car paths are determined.

All blocks with the same destination will share their common arrival station's capacity, no matter what their origins are. Similarly, all blocks with the same origin will occupy a common departure station's sort tracks, no matter what their destinations are. Consequently, the number of trains that can arrive at and can be formed in each station are indirectly limited by that station's classification capacity and number of sort tracks, respectively.

Next to being a strict regulation, the intree rule also ensures that any feasible solution of a TFP has an *arborescence* structure. This means that for each destination separately, a solution specifies a set of directed arcs (blocks), that correspond to a directed tree rooted at the destination. We illustrate this by an example. In Figure 2, assume that demands $1 \to 4$ and $5 \to 4$ meet at 2. The intree rule then requires them to be handled in an identical way from 2 to 4, together with the demand $2 \to 4$. Another reclassification at 3 is needed if the block $2 \to 4$ is not available, see Figure 2(b); otherwise these demands pass straight through 3, as shown Figure 2(c). In either case, the solution for the five demands with destination 4 forms a directed tree. This also illustrates the premise that paths of demands with the same destination must form an undirected tree, e.g., the car paths use shortest paths. We explain this by giving a counter example. Assume that the paths for the demands $1 \to 4$ and $5 \to 4$ are given by $\{1, 2, 5, 6, 3, 4\}$ and $\{5, 2, 3, 4\}$, respectively. If these cars are classified at 2, this inevitably will lead to a contradiction: Neither the path $\{2, 5, 6, 3, 4\}$ nor $\{2, 3, 4\}$ is consistent with the paths of both demands that are given as input. This observation inspires us to decompose the whole network into a series of trees for each destination separately. This decomposition is the fundamental starting point of our solution method. Furthermore, each tree can

be decomposed further into some manageable subtrees. For more details, please see §5.1.

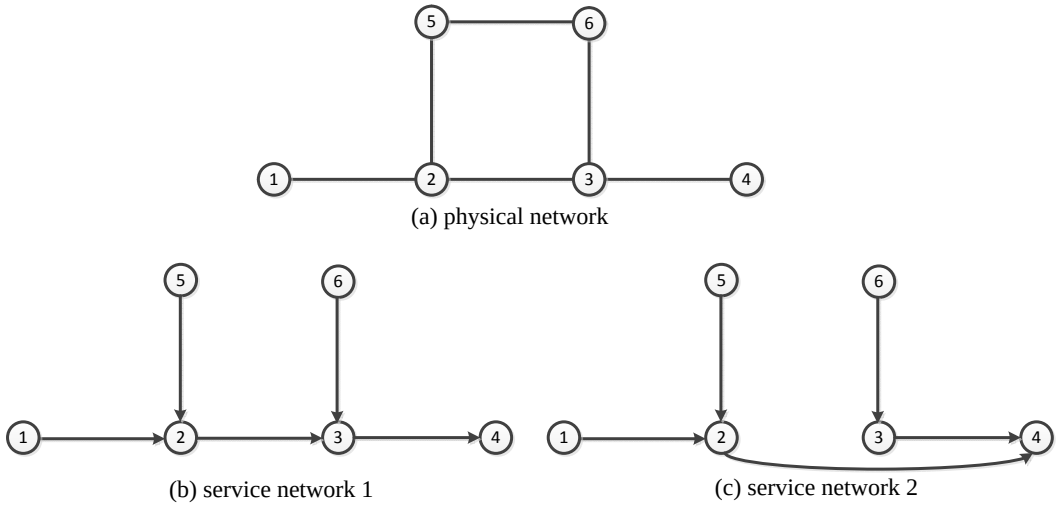


Figure 2: Physical network and two possible service networks

Summarizing, the one-block train formation problem addresses the block design and the car-to-block assignment and aims at minimizing the train costs and delay costs in the tactical level. Next to the classification capacity and sort tracks limitation on each station, both the unitary rule and the intree rule have to be incorporated as well. In essence, the problem could be viewed as a tree-based service network design problem.

3 Literature review

In the past few decades, a massive amount of innovative optimization models and solution algorithms have been proposed in the literature to tackle the train formation problem in railway freight transportation. Readers may refer to Cordeau et al. (1998), Ahuja et al. (2005) and Gorman et al. (2014) for comprehensive surveys. In the following, we first review the literature on blocking, then on train service network design, and finally on the combination of the TFP and train scheduling.

Bodin et al. (1980), Assad (1983), Newton et al. (1998), Barnhart et al. (2000), Ahuja et al. (2007), Yaghini et al. (2011) and Lin et al. (2012) study both block design and car-to-block assignment. In most of them, mixed integer programming (MIP) models are presented in which block services are modeled as 0-1 variables and traffic distributions by continuous variables. Bodin et al. (1980) suggested piecewise linear delay cost with flow balance, yard capacity, block formation, and block capacity constraints. Assad (1983) proposed a shortest path model and dynamic programming method for a linear network. Newton et al. (1998) and Barnhart et al. (2000) formulated a service network design model with node-budget constraints and node-flow constraints which is solved by branch-and-price and Lagrangian relaxation, respectively. Ahuja et al. (2007) developed a very large-scale neighborhood (VLSN) search algorithm to solve real-life railroad blocking problems. Yaghini et al. (2011) built a binary program and applied an ant colony optimization algorithm to instances of Iran Railways. Lin et al. (2012) formulated a bi-level model in which the upper level

determines train connection services, and the lower level assigns a sequence of train services to each shipment. They employed a simulated annealing method to optimize the model. With respect to the CBA variables, both Newton et al. (1998) and Barnhart et al. (2000) use path-based models equipped with convexity constraint to guarantee that each commodity reaches its destination. The remaining ones propose arc-based models in which the flow balance is the backbone constraint. With respect to the objective function, Ahuja et al. (2007) aims at classification cost and flow cost in blocks, whereas flow cost in blocks is the sole component in Newton et al. (1998), Barnhart et al. (2000) and Yaghini et al. (2011).

We now discuss the literature on train service network design. There have been fruitful achievements for this problem, that determines the origin, destination, routing, frequency, makeup and block swap operations for individual trains. Generally speaking, the blocks may either be determined endogenously or be given as a fixed input. In most of the models, either 0-1 variables indicate whether train services are provided, or integer variables indicate train frequencies. The objective function considers train cost (e.g., locomotive powers, hauling, crew), train and car operating cost (e.g., transfer, assembly, disassembly), travel time on the path, and others. One of the first models for the TFP is Assad (1980) who focused on car routing and train makeup. Crainic et al. (1984) dealt with blocking, the train origin, destination, makeup and frequency, and traffic routing based on a general service network design. Keaton (1989) studied car blocking and train routing and makeup. In a subsequent paper by Keaton (1992), the single path constraint and service level for each demand are also included. Martinelli and Teng (1996) simultaneously optimized train routing, blocking, and traffic distribution. Marin and Salmeron (1996a,b) determined the train frequency and car to train assignment. Lulli et al. (2011) studied the traffic distribution and the train frequency problem that arose at an Italian railway company. Fugenschuh et al. (2015) explored a MIP formulation for car routing, train frequency and the assignment of sort tracks to trains for an instance arising at Deutsche Bahn. Khaled et al. (2015) developed the block-to-train assignment and train routing and frequency in a disrupted situation.

Finally, we consider the combination of the TFP with train scheduling, in which the departure and arrival time for each train, the locomotive-to-train assignment, and the crew-to-train assignment are optimized. In most cases, a time-space network model is used that is constructed by replicating the physical network along the time axis. Each yard of the physical network is represented by 2 nodes in each period. One node generates the flow (car, block, train) departing from the yard whereas the other node attracts the flow arriving at the yard. The arcs represent the flow movements such as traveling along a physical link or undergoing operations or waiting in a yard. Two parameters are involved. The first one specifies the time interval between consecutive representations of the same yard. The second one is the length of the planning horizon which is the time represented by the model, also called the schedule length. Haghani (1989) studied train routing and makeup, empty car distribution and timetabling on a one-layer time-space network. Huntley et al. (1995) incorporated train makeup, routing and timetabling. Gorman (1998) studied train service, makeup, and weekly timetables and discretized the time horizon in hours. Jha et al. (2008) addressed the daily version of the BTA problem when given blocks, car-to-block assignments and a train network. Jin et al. (2013) investigated train routing, BTA and crew-to-train assignment decisions based on a problem

solving competition organized by the INFORMS Railway Application Section in 2011. Zhu et al. (2014) integrated blocking, traffic distribution, train makeup and routing, and timetabling on a cyclic three-layer time-space network.

To the best knowledge of the authors, almost none of the current contributions integrates the intree rule into the train formation problem. Among the few that do, Bodin et al. (1980), Jarrah et al. (2009) and Fugenschuh et al. (2015) address problems that are different from the train formation problem in the Chinese railway background. Only Lin et al. (2012) do consider the intree rule for the Chinese background. They use decision variables in a recursive form which inevitably leads to polynomials of high degree in the formulation. In contrast, a linear binary program and a novel tree-based decomposition algorithm are proposed in this paper. The next two sections detail our mathematical formulation and solution algorithm.

4 Mathematical formulation

In this section, we develop a linear binary program for the one-block train formation problem. The aim is to model decisions on block design and car-to-block assignment. The objective is the minimization of the total sum of train cost and classification delay in the tactical level, subject to basic network design constraints and specific requirements based on the Chinese railway background.

4.1 Parameters and variables

We first list the parameters defining the physical network and the demand in Table 3. Table 4 lists two kinds of 0-1 decision variables that are used in the one-block TFP model.

Table 3: Parameters of the one-block TFP model

Notation	Definition	Unit	Remark
V	set of stations, $\|V\| = N$		index i, j, o, d
E	set of links		index $e = (i, j)$
α_i	classification capacity in station i, $\forall i \in V$	car	constant
ϕ_i	reserve ratio of classification capacity in station i, $\forall i \in V$		constant
L	parking capacity on each sort track	car	constant
β_i	number of sort tracks in station i, $\forall i \in V$		constant
φ_i	reserve ratio of number of sort tracks in station i, $\forall i \in V$		constant
t_i	classification delay time per car in station i, $\forall i \in V$	hour	constant
$c_{i,j}$	accumulation parameter for train from i to j, $\forall i, j \in V$	hour	constant
$m_{i,j}$	size of train from i to j, $\forall i, j \in V$	car	constant
$f_{i,j}$	traffic volume of demand from i to j, $\forall i, j \in V$	car	constant
$P_{i,j}$	path for demand from i to j, $\forall i, j \in V$		$\overline{P}_{i,j} = P_{i,j} - \{i, j\}$
$g(k; i, j)$	position of station k on path $P_{i,j}$, $\forall i, j \in V, k \in P_{i,j}$		$g(i; i, j) = 1$ $g(j; i, j) = \|P_{i,j}\|$
$B_{i,j}$	block from i to j, $\forall i, j \in V$		
$\Omega_{i,j}$	set of all possible CBA plans for demand from i to j, $\forall i, j \in V$		index $I_{i,j}^{n_1, n_2, \cdots, n_l}$

The one-block TFP is modeled on an undirected graph (V, E), where V denotes the set of all stations, and $E \subseteq V \times V$ the set of links between two adjacent stations. For each demand, also referred to as an OD, the route is given in advance and can be represented as a path through this

Table 4: Decision variables of the one-block train TFP model

Notation	Definition	Remark
$x_{i,j}$	equals 1 if block $B_{i,j}$ is provided, 0 otherwise	block design
$x_{i,j}^{n_1,n_2,\cdots,n_l}$	equals 1 if demand from i to j chooses the CBA $I_{i,j}^{n_1,n_2,\cdots,n_l}$, 0 otherwise	car distribution

graph from the demand's origin to its destination. For the one-block TFP, a block design variable $x_{i,j} = 1$ means that a direct train is provided from station i to station j. This also implies that the OD from station i to j is carried on this direct train without classification. Next to these variables, we also define decision variables that explicitly list all classification stations for a demand from i to j. We refer to these variables as car distribution variables. If a demand from i to j chooses the CBA plan $I_{i,j}^{n_1,n_2,\cdots,n_l}$, this demand will be successively classified at station $n_1, n_2, \cdots, n_l$ along its path and successively assembled into blocks B_{i,n_1}, B_{n_1,n_2},$\cdots$, B_{n_{l-1},n_l}, $B_{n_l,j}$. Intuitively, one could view $I_{i,j}^{n_1,n_2,\cdots,n_l}$ as $\{B_{n_u,n_{u+1}}, u = 0, 1, \cdots, l\}$, where $n_0 = i$, $n_{l+1} = j$.

To be more specific, for each possible CBA plan $I_{i,j}^{n_1,n_2,\cdots,n_l}$ for a demand from i to j, the classification stations must be on the demand's path, i.e., $n_1, n_2, \cdots, n_l \in \overline{P}_{i,j}$. Furthermore, the classification sequence must be consistent with the positions in the path, i.e., $1 < g(n_1; i, j) < g(n_2; i, j) < \cdots < g(n_l; i, j) < |P_{i,j}|$. The set of all possible CBAs could be explicitly defined as

$$\Omega_{i,j} = \{I_{i,j}^{n_1,n_2,\cdots,n_l} | 1 < g(n_1; i, j) < g(n_2; i, j) < \cdots < g(n_l; i, j) < |P_{i,j}|, n_1, n_2, \cdots, n_l \in \overline{P}_{i,j}\} \quad (1)$$

for all $i, j \in V$. For example, using that $P_{1,4} = \{1, 2, 3, 4\}$ for the demand $1 \to 4$ in Figure 2(a), it follows that $\Omega_{1,4} = \{I_{1,4}^2, I_{1,4}^3, I_{1,4}^{2,3}\}$.

The total number of possible CBAs for the demand $i \to j$ is $|\Omega_{i,j}| = \sum_{k=1}^{|\overline{P}_{i,j}|} \binom{|\overline{P}_{i,j}|}{k} = 2^{|\overline{P}_{i,j}|} - 1$. Particularly, $\Omega_{i,j} = \emptyset$ when $(i, j) \in E$. That is to say, there must be a block between two adjacent stations. The set of CBAs can be enumerated in advance. Indeed, its size increases exponentially as the problem size grows. In practice, too many classifications are unfavorable when the contract deadline is guaranteed. We can take this into account by limiting the maximal number of classifications for each demand, loosely estimated from the due dates and the average handling time. We denote this maximum number of classifications as $\eta_{i,j}, \forall i, j \in V$. When we incorporate this, the size of the model is limited to some extent: $|\Omega_{i,j}| = 2^{\min(|\overline{P}_{i,j}|, \eta_{i,j})} - 1$. In this case, our model integrates service quality on behalf of clients and cost-benefit on behalf of the railroads.

According to the selection of CBAs for all demands, the number of cars contained in each block can be determined by the following formula.

$$F_{i,j} = f_{i,j} x_{i,j} + \sum_{o,d \in V} \sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l} \in \Omega_{o,d} | B_{i,j} \in I_{o,d}^{n_1,\cdots,n_l}\}} f_{o,d} x_{o,d}^{n_1,n_2,\cdots,n_l} \quad (2)$$

for all $i, j \in V$. Here, we sum two terms. The first term is the corresponding origin-destination demand itself, while the second term includes all other demands which take this block as a part of their journeys. For example, $F_{2,4} = f_{2,4} + f_{1,4} + f_{5,4}$ for the block design in Figure 2(c).

4.2 0-1 formulation

Using the decision variables and notation defined in the previous section, we can model the one-block train formation problem as follows.

$$\min \sum_{i,j\in V} c_{i,j} m_{i,j} x_{i,j} + \sum_{i,j\in V} \sum_{I_{i,j}^{n_1,n_2,\cdots,n_l}\in\Omega_{i,j}} \sum_{k\in\{n_1,n_2,\cdots,n_l\}} f_{i,j} t_k x_{i,j}^{n_1,n_2,\cdots,n_l} \tag{3}$$

subject to

$$x_{i,j} + \sum_{I_{i,j}^{n_1,n_2,\cdots,n_l}\in\Omega_{i,j}} x_{i,j}^{n_1,n_2,\cdots,n_l} = 1, \qquad \forall i,j \in V \tag{4}$$

$$\sum_{i\in V} F_{i,j} \le \alpha_j(1-\phi_j), \qquad \forall j \in V \tag{5}$$

$$\sum_{j\in V} F_{i,j} \le L\beta_i(1-\varphi_i), \qquad \forall i \in V \tag{6}$$

$$x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{n_u,n_{u+1}}, \qquad \forall i,j \in V, I_{i,j}^{n_1,n_2,\cdots,n_l} \in \Omega_{i,j}, u = 0,1,\cdots,l \tag{7}$$

$$x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{n_1,j}^{n_2,\cdots,n_l}, \qquad \forall i,j \in V, I_{i,j}^{n_1,n_2,\cdots,n_l} \in \Omega_{i,j}, l \ge 2 \tag{8}$$

$$x_{i,j} \in \{0,1\}, \qquad \forall i,j \in V \tag{9}$$

$$x_{i,j}^{n_1,n_2,\cdots,n_l} \in \{0,1\}, \qquad \forall i,j \in V, I_{i,j}^{n_1,n_2,\cdots,n_l} \in \Omega_{i,j} \tag{10}$$

The objective function (3) minimizes the total cost in the tactical level. The first term represents the train cost that can be regarded as fixed design cost. The second term represents the delay cost that can be regarded as variable operating cost.

Equality (4) implements the unitary rule. For each demand, we select one and only one opportunity: either a direct block is provided, or the demand is classified at one or more intermediate stations. The former means that the demand will be delivered by a direct train, instead of by a series of connected trains for the latter. At the same time, this constraint guarantees that every car can reach its destination.

Constraint (5) ensures that the amount of cars classified at each station does not exceed that station's total capacity. This constraint is analogous to the yard capacity equation in Bodin et al. (1980), or the node-flow constraint in Newton et al. (1998) and Barnhart et al. (2000). According to formula (2), the left hand side can be rewritten as

$$\sum_{i\in V} F_{i,j} = \sum_{i\in V} f_{i,j} + \sum_{o,d\in V} \sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l}\in\Omega_{o,d} | j\in\{n_1,\cdots,n_l\}\}} f_{o,d} x_{o,d}^{n_1,n_2,\cdots,n_l}, \forall j \in V.$$

In this equation, the constant term is the sum of volumes of demands that have this station as destination. The second term only includes the CBA variables of those demands that consume this station's capacity on their intermediate stop(s). It follows that the classification capacity constraints

are equivalent to

$$\sum_{o,d\in V}\sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l}\in\Omega_{o,d}|j\in\{n_1,\cdots,n_l\}\}} f_{o,d}x_{o,d}^{n_1,n_2,\cdots,n_l} \le \alpha_j(1-\phi_j) - \sum_{i\in V} f_{i,j}, \forall j \in V. \tag{11}$$

Similarly, inequality (6) ensures that the amount of cars in all blocks that depart from a station divided by the tracks' parking capacity is at most the number of sort tracks in this station. This constraint is similar to the block capacity constraint in Bodin et al. (1980), or the separate node-budget constraint in Newton et al. (1998) and Barnhart et al. (2000). Using again formula (2), the left hand side can be written as

$$\sum_{j\in V} F_{i,j} = \sum_{j\in V} f_{i,j} + \sum_{o,d\in V}\sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l}\in\Omega_{o,d}|i\in\{n_1,\cdots,n_l\}\}} f_{o,d}x_{o,d}^{n_1,n_2,\cdots,n_l}, \forall i \in V.$$

Here, the constant term is the sum of volumes of demands that originate from this station. The second term only includes the CBA variables of those demands that occupy this station's sort tracks on their intermediate stop(s). This shows that the sort track constraints are equivalent to

$$\sum_{o,d\in V}\sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l}\in\Omega_{o,d}|i\in\{n_1,\cdots,n_l\}\}} f_{o,d}x_{o,d}^{n_1,n_2,\cdots,n_l} \le L\beta_i(1-\varphi_i) - \sum_{j\in V} f_{i,j}, \forall i \in V. \tag{12}$$

Inequalities (7) are cross linking constraints between the two types of variables. Only if all related blocks which successively connect the origin, classification station(s), and destination of a demand are available, the corresponding CBA can be chosen. Inequalities (8) model the intree rule. If cars from station i to j are classified at station $n_1, n_2, \cdots, n_l$ in succession, i.e., $x_{i,j}^{n_1,n_2,\cdots,n_l} = 1$, then cars from n_1 to j must be classified at $n_2, \cdots, n_l$, i.e., $x_{n_1,j}^{n_2,\cdots,n_l} = 1$. Note that $x_{n_1,j}^{n_2,\cdots,n_l}$ is actually a block $x_{n_1,j}$ when $l = 1$. Therefore, once classified, the demand has exactly the same classification policy as the demand that originates from this station with the same destination.

There are some redundant inequalities for constraint (7). For example, $x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{n_1,j}^{n_2,\cdots,n_l}$ together with $x_{n_1,j}^{n_2,\cdots,n_l} \le x_{n_1,n_2}$ implicitly covers the cross linking relationship $x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{n_1,n_2}$. In order to prevent this, constraints (7) and (8) can be expressed compactly as

$$\begin{cases} x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{i,n_1},\ x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{n_1,j}, & \text{if } l = 1 \\ x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{i,n_1},\ x_{i,j}^{n_1,n_2,\cdots,n_l} \le x_{n_1,j}^{n_2,\cdots,n_l}, & \text{if } l > 1 \end{cases} \tag{13}$$

for all $i, j \in V, I_{i,j}^{n_1,n_2,\cdots,n_l} \in \Omega_{i,j}$. In this way, there are exactly two constraints for each CBA variable.

Both constraint (9) and (10) specify the domain of decision variables.

4.3 Characteristic analysis

The model (3)-(4), (9)-(13) is a linear binary program. In this model, we do not take the capacity of the arcs (blocks) into consideration, in contrast to conventional models for multi-commodity capacitated network design problems (Gendron et al. 1999). Instead, in this paper, both classification capacity and sort track constraints are incorporated to restrict too many blocks originating

from or destined to any station. The size of the model could be roughly estimated: the model has $N(N-1)2^{N-2}$ variables and $2N(N-1)2^{N-2} - N^2 + 3N$ constraints. The model scale increases exponentially as the problem size grows.

A comparison between models from literature and the one used in this paper is presented in Table 5. The intree requirement may be the greatest difference in the field of railway freight transportation between China and other countries. In most literature, for example in Lin et al. (2012), the model decides for every OD where it is classified first. Using these recursive decision variables, the intree rule is automatically satisfied. Unlike this, we define novel decision variables that list all classification station(s) for a demand and provide a class of linear constraints to enforce the intree rule. Obviously, the former model is recursive and implicit; while the model in this paper is explicit and thus allows to be solved exactly. The resulting values of the two types of decision variables could be transformed mutually. Because of the correspondence between blocks and trains in the one-train TFP, a block could be naturally regarded as a local service arc as well. However, car classification variables in this paper indicate how to operate a demand along its complete path though the network. This means that existing models could be marked as arc-based, while this paper is path-based in the sight of model structure.

Table 5: Comparison between literature and this paper

	Literature	This paper
Meet intree requirement	decision variable	constraint
Classification variable	recursive, implicit	enumerative, explicit
Model structure	non-linear, arc-based	linear, path-based

A coin has two sides. The disadvantage of an enumerative model as developed in this paper is the huge numbers of variables and constraints in the model. The advantage is an elegant linear model. On the contrary, earlier approaches inevitably lead to polynomials of high degree in explicit expressions. We take the example in Figure 1 to explain this. In order to avoid ambiguity, denote the CBA variable used in earlier models by $y^k_{i,j}$. This variable takes value 1 if the OD from i to j is firstly classified at k, and 0 otherwise, $\forall i, j \in V, k \in \overline{P}_{i,j}$. In terms of these variables, the total classification cost can be expressed as $f_{1,3}t_2y^2_{1,3} + f_{2,4}t_3y^3_{2,4} + f_{1,4}(t_2y^2_{1,4} + t_3y^3_{1,4} + t_3y^2_{1,4}y^3_{2,4})$. This polynomial gives rise to a non-linear objective. The larger the size of the network, the higher the powers of the terms. Polynomial terms can be linearized by introducing new variables and constraints. However, the price to be paid is heavy, such as the need to introduce a huge amount of additional variables and constraints, and complicated formal expressions in a general situation.

5 Tree-based decomposition algorithm

In the previous section, we presented a mathematical model for the one-block train formation problem. This model contains an exponential number of variables and constraints. As explained in §2, for each destination separately, the intree rule does not only require the car paths to form an undirected tree, but also specifies the solution of the TFP to form a directed tree. Based on these arborescence structures, the original problem can be viewed as the addition of several subproblems which

correspond to separable trees rooted at each destination. We now first explain how these trees can be decomposed further into some manageable subtrees. A restricted model is then derived to find the optimal solution for each subtree. While solving subtrees sequentially, we make sure that the solutions of the subproblems can be combined into a global solution for the holistic network. Finally, two model enforcements are discussed that accelerate the solution process. Our overall solution methodology is shown in the flowchart depicted in Figure 3.

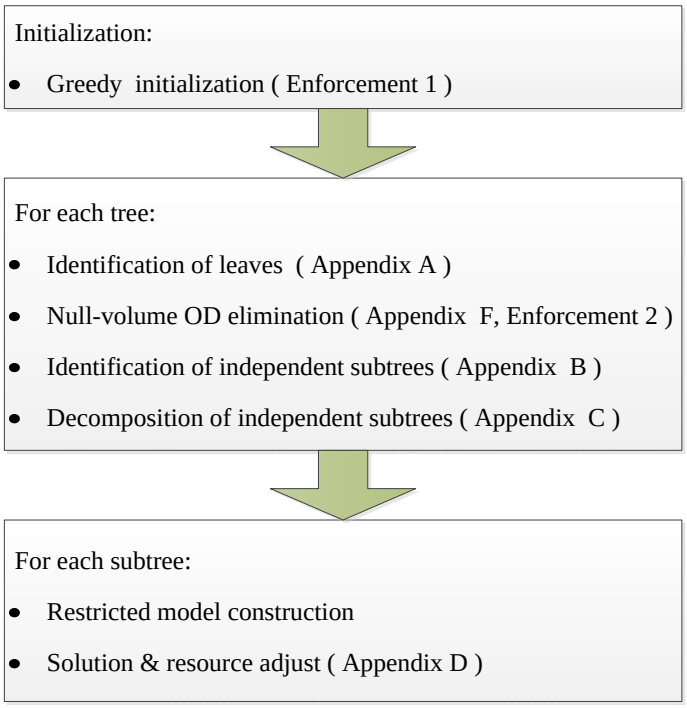


Figure 3: Overall Flowchart

5.1 Tree-based decomposition

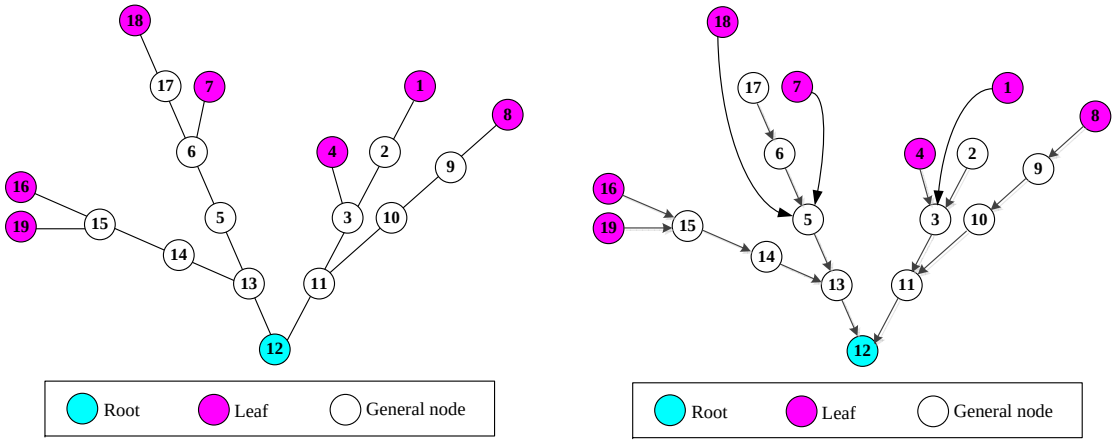
The TFP for a given destination (a rooted tree) is decomposed into subtrees in two stages. In the first stage, the tree for a given destination is decomposed into some independent subtrees. If these independent subtrees are still too large to be solved, they will be divided into smaller subtrees in the second stage. The size of the subtrees is regulated by a node size parameter that can balance computation time and solution quality.

5.1.1 Identification of independent subtrees

In order to explain our decomposition approach, we first define some concepts from graph theory. Consider a graph $G = (V, E)$. A node $u \in V$ is a *neighbor* of $v \in V$ if $\{u, v\} \in E$. The *degree* of a vertex $v \in V$ is the number of edges incident to that vertex. A *leaf* in the graph is a vertex with degree 1. A *path* in the graph is a finite sequence of different vertices $(v_0, v_1, \ldots, v_n)$ with the property that $\{v_{i-1}, v_i\} \in E$ for all $i = 1, \ldots, n$. The *length* of a path is the number of vertices it contains. A *tree* in G is a subgraph of G with the property that every pair of vertices is connected

by exactly one path. If one vertex in a tree is specified as the *root* of the tree, we refer it as a *rooted tree*. Finally, we define a *branch* in a rooted tree as a path from a leaf to the root. For convenience, we denote a branch in this tree by the pair $\langle o, d \rangle$ where o is a leaf and d is the root node.

The intree rule now requires that for each destination d, both the car paths and the solution of the TFP can be viewed as a tree rooted at d. The difference is that the former is an undirected tree whereas the latter is a directed tree. For example, Figure 4(a) gives shortest paths for all demands with destination 12 for the small case in Lin et al. (2012). Figure 4(b) represents the optimal train formation plan concerned. Figure 4(a) has 7 leaves and 7 branches. For the sake of clarity, the leaves and branches are listed in Table 6.



(a) Car routes rooted at 12

(b) Feasible TFP rooted at 12

Figure 4: Two rooted trees

Table 6: Leaves and branches of the tree rooted at 12 in Figure 4(a)

No.	Leaf	Branch
1	19	19-15-14-13-12
2	16	16-15-14-13-12
3	18	18-17-6-5-13-12
4	7	7-6-5-13-12
5	4	4-3-11-12
6	1	1-2-3-11-12
7	8	8-9-10-11-12

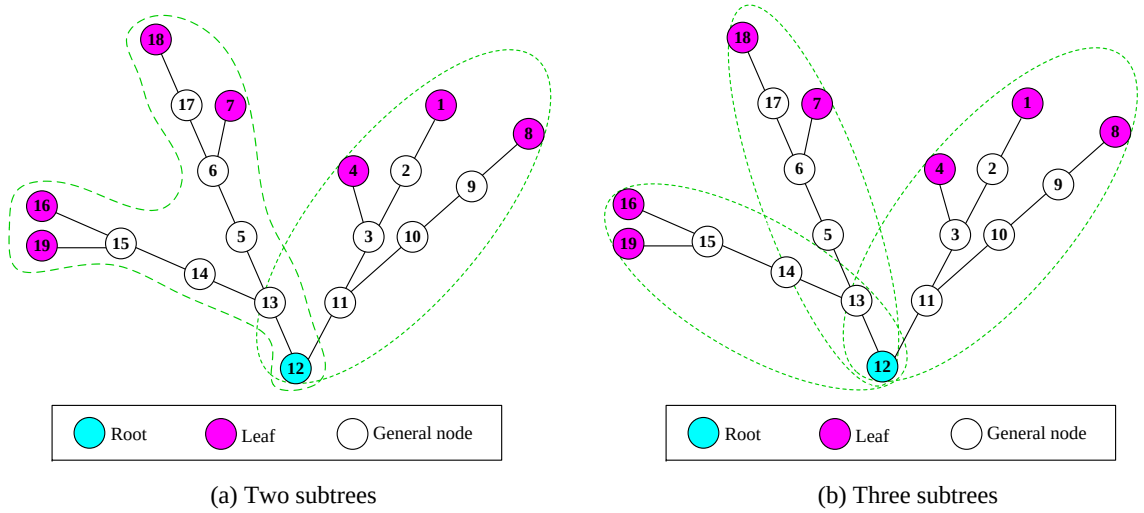
We now give the definition of a subtree. It follows from this definition that a subtree is fully characterized by the leaves it contains. Appendix A describes how to identify leaves in a rooted tree.

Definition 1. *In a rooted tree, a* subtree *is the union of a subset of its branches. Particularly, two subtrees that have no vertices in common except the root are called* independent *subtrees.*

Two independent subtrees in a rooted tree have no common edges. There are several ways to divide a tree into subtrees. However, as the following proposition explains, a division into independent subtrees is favorable.

Proposition 1. *Let a rooted tree be decomposed into a set of mutually independent subtrees. The optimal solution in this rooted tree can be found by solving the independent subtrees separately.*

Two subtrees are *not* independent if they share a common vertex besides the root. In this case, both subtrees contain a path towards the destination that includes this common vertex. Given that the path from this common vertex to the root is unique, it follows that the subtrees also contain the same neighbor of the root. This implies that every neighbor of the root defines an independent subtree. This is the key idea of the algorithm in Appendix B. Taking Figure 4(a) for example, the neighbors of the root are 13 and 11. Two independent subtrees are identified, shown in Figure 5(a). The first subtree $\{\langle 19, 12\rangle, \langle 16, 12\rangle, \langle 18, 12\rangle, \langle 7, 12\rangle\}$ is the union of the first four branches in Table 6. The second subtree $\{\langle 4, 12\rangle, \langle 1, 12\rangle, \langle 8, 12\rangle\}$ is the union of the last three branches in Table 6.



(a) Two subtrees (b) Three subtrees

Figure 5: Decomposition in subtrees

5.1.2 Decomposition of independent subtrees

Independent subtrees are smaller than the complete tree for a given destination. However, for large-scale networks, the independent subtrees can still be fairly large. Therefore, we decompose the subtrees further. Of course, if an independent subtree is decomposed further, the resulting subtrees might not be independent anymore.

In order to control the size of the resulting subtrees, we introduce the *node size parameter*, i.e., the maximum number of vertices allowed in each subtree. We denote it by K, where $K \leq N$. The value of K is required to be at least the length of the *longest* branch of the rooted tree. Table 7 compares the sizes of a network, tree and subtree in three aspects.

We now explain how to further decompose a set of independent subtrees. For more details, see the pseudo-code in Appendix C. We propose a recursive procedure, that decomposes a given subtree into smaller ones if it includes more than K nodes. This recursive procedure will not terminate until all resulting subtrees include at most K nodes. In this case, the subtrees are fully decomposed. In order to introduce this procedure, we first define a bifurcation node and critical nodes. In these

Table 7: Comparison of the size of a network, tree, and subtree

	Network	Tree	Subtree
Nodes	N	N	$\leq K$
Links	$\|E\|$	$N-1$	$\leq K-1$
Demands	$N(N-1)$	$N-1$	$\leq K-1$

definitions, we assume that a given rooted tree has been decomposed in independent subtrees as explained before. This implies that the root in each subtree has degree 1.

Definition 2. *In a rooted subtree, the* bifurcation *node is the node that is* closest *to the root with a degree at least 3.*

Definition 3. *In a rooted subtree, the* critical *nodes are the* farthest *neighbors of the bifurcation node.*

We prove the existence and uniqueness of the bifurcation node in Appendix D. If the node size of a subtree exceeds the specified upper bound, it is subdivided into smaller subtrees according to its critical nodes, i.e., each critical node gives rise to one subtree. The subtree shrinks as the bifurcation node approaches the leaves. Taking Figure 5(a) as an example, given $K = 10$, only the subtree $\{\langle 19, 12\rangle, \langle 16, 12\rangle, \langle 18, 12\rangle, \langle 7, 12\rangle\}$ with 11 nodes needs to be divided into smaller ones. This subtree has a bifurcation node 13. Among its three neighbors 14, 5, and 12, only the first two are critical nodes. Accordingly, we obtain a subtree $\{\langle 19, 12\rangle, \langle 16, 12\rangle\}$ for the critical node 14, and another subtree $\{\langle 18, 12\rangle, \langle 7, 12\rangle\}$ for the critical node 5, see Figure 5(b). The two resulting subtrees have 6 and 7 nodes respectively. Because the resulting subtrees are now small enough, the decomposition procedure terminates. Note that for $K \geq 11$, the original subtrees would not be further decomposed. In that case, the subtrees remain independent, as depicted in Figure 5(a).

5.2 Restricted model

After the network has been decomposed into subtrees, a restricted model is constructed to optimize each subtree separately. As part of the parameter tuning in §6.2.2, we analyze the impact of the order in which the subtrees are solved. When solving each subtree, we make sure that a feasible solution is available for the remaining subtrees. We now discuss the decision variables, cost coefficients, input parameters and constraints in this restricted model.

5.2.1 Decision variables

Essentially, every decision variable corresponds to a decision on transporting demand between a pair of stations. If we focus on a subtree, we include variables for all pairs of nodes on all branches of the subtree. The demands to be transported to the common root are definitely included. In addition, we also include the pairs from and/or to intermediate nodes of each branch. This must be done because of the cross linking constraints between block design variables and CBA variables. We use the definition of a path for each demand as a sequence from its origin to its destination and the corresponding notation here as well. All paths are oriented along the direction from the leaf towards

the root. Because only a subset of the nodes are considered, the number of variables and constraints in the model becomes smaller for subtrees.

We are going to solve the subtrees sequentially rather than simultaneously. As a consequence, when optimizing a subtree, some variables may already be set when the restricted model for other subtrees was solved. For instance, in Figure 5(b) both the subtree $\{\langle 19, 12\rangle, \langle 16, 12\rangle\}$ and $\{\langle 18, 12\rangle, \langle 7, 12\rangle\}$ share a common section 13-12. After solving the first subtree, the decision for the demand from 13 to 12 is set. This decision must be incorporated when the second subtree is solved. We now explain how to deal with this. For a given pair of stations i, j, there are 3 options.

- case 1: Neither $x_{i,j}$ nor $x_{i,j}^{n_1,n_2,\cdots,n_l}, \forall I_{i,j}^{n_1,n_2,\cdots,n_l} \in \Omega_{i,j}$ has been determined;
- case 2: $x_{i,j} = 1$, $x_{i,j}^{n_1,n_2,\cdots,n_l} = 0, \forall I_{i,j}^{n_1,n_2,\cdots,n_l} \in \Omega_{i,j}$;
- case 3: $x_{i,j} = 0$, $x_{i,j}^{n_1,n_2,\cdots,n_l} = 1$ for some $I_{i,j}^{n_1,n_2,\cdots,n_l} \in \Omega_{i,j}$.

For the first case, neither the block service nor any possible CBA has been selected. In the second case, a direct block from i to j has been selected. In the third case, a CBA is selected for the demand from i to j. Recall that the intree constraints (13) are enforced for each car-to-block assignment variable. As a result, other variables are forced to change when a CBA variable changes value. In particular, all variables related to $x_{i,j}^{n_1,n_2,\cdots,n_l}$ are included in a chain

$$\cdots \leq x_{o,j}^{i,n_1,n_2,\cdots,n_l} \leq x_{i,j}^{n_1,n_2,\cdots,n_l} \leq x_{n_1,j}^{n_2,\cdots,n_l} \leq \cdots \leq x_{n_l,j}.$$

All *predecessors* in this chain, located to the left of $x_{i,j}^{n_1,n_2,\cdots,n_l}$, are set to 0 when $x_{i,j}^{n_1,n_2,\cdots,n_l} = 0$. On the contrary, all *successors* in the chain, located to the right of $x_{i,j}^{n_1,n_2,\cdots,n_l}$, have to be set to 1 when $x_{i,j}^{n_1,n_2,\cdots,n_l} = 1$. In order to ensure that we can control such propagation, we apply the following procedure when generating the restricted model. In the first case, all variables for the pair i, j are included in the model. In the second case, we only include the variable $x_{i,j}$, which is then forced to 1 by constraint (4). Finally in the third case, we include $x_{i,j}$ and the one CBA variable $x_{i,j}^{n_1,n_2,\cdots,n_l}$ that has been selected when solving an earlier subtree. In short, we allow to replace a CBA by a direct block in later iterations, but not the other way around. Therefore, we refer to block variables as *lazy* variables, to CBA variables as *active* variables, and to this strategy as an *active-lazy-strategy*.

5.2.2 Cost coefficients

In the previous paragraph, we have discussed the variables that are present in a restricted model. We now explain how the cost coefficients are defined. For the first two cases in §5.2.1, the cost coefficients are equal to those in §4.2. For the third case in §5.2.1, the objective coefficients are adapted.

In case 3, $x_{i,j}^{n_1,n_2,\cdots,n_l}$ is set to 1 in a previous iteration. If this variable is kept unchanged, no additional cost are incurred, as these costs have already been included in the previous iteration. On the other hand, if $x_{i,j}$ is set to 1, additional train costs are incurred, but delay costs are saved for all demands that were previously classified between i and j. This benefit is taken into account in the restricted model. There are 4 possibilities when a new block $B_{i,j}$ replaces $I_{i,j}^{n_1,n_2,\cdots,n_l}$, which are

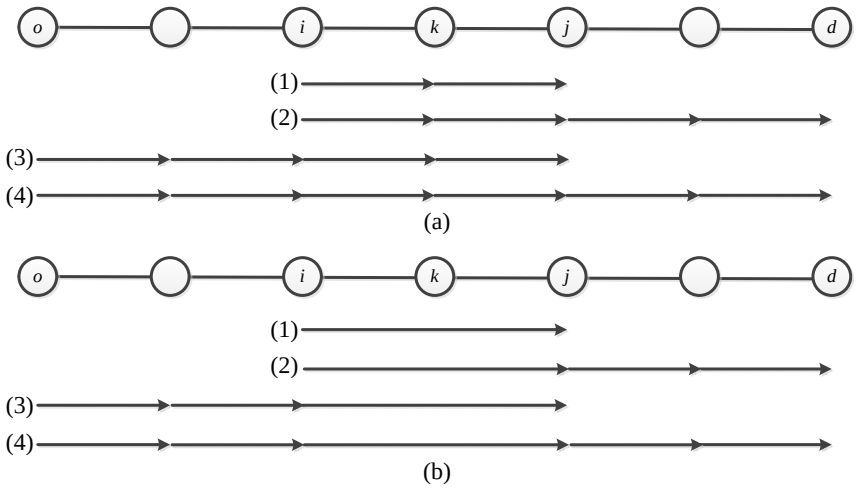


Figure 6: Classification cost saved

depicted in Figure 6(a). In Figure 6(b), the classification policies are depicted after block $B_{i,j}$ has been selected.

In all 4 possibilities, the classifications between stations i and j are not performed anymore. Therefore, delay costs are reduced in this situation. The delay reduction in the first case can be regarded as *direct benefit* and the delay reduction for the others as *indirect benefit*. This reduction in delay cost is incorporated in the cost coefficient of $x_{i,j}$. In order to quantify the reduction in delay cost, define $\widetilde{P}_{o,d}$ as an ordered set of the origin, successive intermediate classification stations, and destination for the OD pair o, d. It holds that $\widetilde{P}_{o,d} \subseteqq P_{o,d}$. The cost coefficient for $x_{i,j}$ is given by

$$c_{i,j}m_{i,j} - \sum_{\{o,d \in V | i,j \in \widetilde{P}_{o,d},\, g(i;o,d) < g(j;o,d)\}} \sum_{k \in \widetilde{P}_{o,d} \bigcap \overline{P}_{i,j}} f_{o,d}t_k. \tag{14}$$

5.2.3 Resource reserve and release

The original model is confronted with the global network and all OD-pairs at once. In the restricted models, we only consider a partial network and a subset of the demands. As a consequence, the resource parameters need to be updated after a restricted model is solved. When solving the models, we should ensure that feasible solutions are available in later iterations as well.

Recall that even direct blocks require sort tracks at the origin and classification capacity at the destination station. In addition, CBA variables require these resources in intermediate stations, too. According to formula (11), all demands will definitely consume classification capacity at their destination stations, no matter whether a direct block or a CBA is selected. Thus, the sum of the volumes of all demands with the same destination can be reserved in that station beforehand. The initial available capacity is then replaced by

$$\gamma_j \leftarrow \alpha_j(1 - \phi_j) - \sum_{i \in V} f_{i,j}, \forall j \in V. \tag{15}$$

Similarly, according to formula (12), all ODs will certainly depart from the sort tracks of their

origination stations, no matter whether a direct block or a CBA is chosen. Thus, the initial number of available tracks is replaced by

$$\delta_i \leftarrow L\beta_i(1-\varphi_i) - \sum_{j\in V} f_{i,j}, \forall i \in V. \tag{16}$$

Special attention should be paid to the CBA variables in case 3 in §5.2.1. When a CBA variable is changed to 0, the resources previously occupied can be released. This occurs, for example, at station k in Figure 6. When a block $B_{i,j}$ is selected instead of the CBA plan $I_{i,j}^{n_1,n_2,\cdots,n_l}$, available resources in station $k \in \{n_1, n_2, \cdots, n_l\}$ are released as

$$\gamma_k \leftarrow \gamma_k + f_{i,j}, \tag{17}$$

$$\delta_k \leftarrow \delta_k + f_{i,j}. \tag{18}$$

Here we only release the resource utilization corresponding to the direct benefit. The reason is that we introduced decision variables in the restricted model only for the first situation in Figure 6(a). As explained in §5.2.1, decision variables are not included for the other three situations. Once a restricted model is solved, the resource utilization corresponding to the indirect benefit will be released. This will be explained in detail in §5.2.5.

5.2.4 Constraints

The constraints in the restricted model are not completely identical to those in §4.2, mainly because the decision variables for case 3 in §5.2.1 are different.

Firstly, we consider the scenario uniqueness constraint. In case 3, all other associated CBA variables are discarded in order to prevent having to propagate the changes. In this situation, only the selected CBA variable itself and a candidate block are included. The corresponding constraint reads

$$x_{i,j} + x_{i,j}^{n_1,n_2,\cdots,n_l} = 1. \tag{19}$$

After reserving the resources, demands only contribute to the classification workload and track usage on intermediate station(s). Thus, the resource constraints (5) and (6) are modified as follows

$$\sum_{o,d\in V} \sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l} \in \Omega_{o,d} | j\in\{n_1,\cdots,n_l\}\}} f_{o,d} x_{o,d}^{n_1,n_2,\cdots,n_l} \le \gamma_j, \forall j \in V, \tag{20}$$

$$\sum_{o,d\in V} \sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l} \in \Omega_{o,d} | i\in\{n_1,\cdots,n_l\}\}} f_{o,d} x_{o,d}^{n_1,n_2,\cdots,n_l} \le \delta_i, \forall i \in V, \tag{21}$$

where γ_j and δ_i are computed using formulas (15) and (17), and formulas (16) and (18), respectively.

There are two variants for constraints (13) due to the absence of some CBA variables. For any $x_{i,j}^{n_1,n_2,\cdots,n_l}$ included in the restricted model, we have the following constraints.

- If the associated block x_{i,n_1} has not been selected in an earlier iteration, then $x_{i,j}^{n_1,n_2,\cdots,n_l} \leq x_{i,n_1}$ has to be included. Otherwise, if x_{i,n_1} has been selected, this inequality holds automatically.
- The intree constraint considers the variables related to the OD from n_1 to j,
  - when $l = 1$, if $x_{n_1,j}$ has not been decided yet, then $x_{i,j}^{n_1} \leq x_{n_1,j}$ has to be included; otherwise, this inequality is not necessary.
  - when $l > 1$, if $x_{n_1,j}^{n_2,\cdots,n_l}$ also is a variable in the restricted model, then $x_{i,j}^{n_1,n_2,\cdots,n_l} \leq x_{n_1,j}^{n_2,\cdots,n_l}$; if another CBA plan rather than $I_{n_1,j}^{n_2,\cdots,n_l}$ has been selected, then $x_{i,j}^{n_1,n_2,\cdots,n_l} = 0$.

5.2.5 Solution and resource adjustment

When a restricted model is solved, its solution should be recorded. Given the decisions that are made, the consumption of resources should be updated as well.

As a consequence of the active-lazy strategy on the decision variables, a CBA plan could be replaced by an associated block for a certain OD. This change does not only reduce the objective function, but also increases the resource availability, because some resources are now not occupied anymore. We have released the resources for the direct benefit in §5.2.3 before solving a restricted model. Next to this direct effect, the CBA might change for some other ODs as well, as explained in §5.2.2. We have referred to this as the indirect benefit of changing a CBA to a direct block, as depicted by the last three possibilities in Figure 6(b). Consequently, the quantities of available resources should be updated. First of all, the change of the solutions should be recorded. Second, previously allocated resource consumption is supposed to be adjusted. Appendix E presents the main steps and it will be called each time we solve a restricted model. Here, the variable H saves the origin-destination, whereas W saves intermediate stops. These are empty at the beginning. After all subtrees are solved, these variables constitute a feasible solution for the whole network.

5.3 Algorithm enforcements

In this section, we discuss two enforcements to improve the performance of the algorithm. A greedy method is applied to fix some variables in the solution without optimization. Then, by eliminating ODs with null-volume, the number of variables could be reduced.

5.3.1 Greedy initialization

The objective function (3) of the one-block train formation model is a sum of train cost and classification cost. The latter is immediately affected by the traffic volumes. If some demand by itself is already large enough, even the minimum classification cost among all CBA plans is larger than the train cost associated with dispatching a direct service, i.e.,

$$\min\{f_{i,j}t_k, k \in \overline{P}_{i,j}\} \geq c_{i,j}m_{i,j}.$$

In this situation, the optimal choice is a direct block rather than classification at any midway station. We apply the above criterion referred to as a *sufficient condition* in Lin et al. (2012) to decide which blocks will be provided beforehand without optimization. This criterion demonstrates the role of blocks as a concept allowing aggregation of dispersive demands with low volume into trains of an appropriate size.

5.3.2 Null-volume OD elimination

Demands with null-volume are special because their classification costs are zero for every CBA plan. Nonetheless, it is not always beneficial to assign a CBA instead of a direct block to a null-volume demand. Taking Figure 2 as an example, let $f_{2,4} = 0$. It is possible that the best service network is that as Figure 2(c), that is, $B_{2,4}$, $I^2_{1,4}$, and $I^2_{5,4}$ are provided for OD $2 \to 4$, $1 \to 4$, and $5 \to 4$ respectively. Here, the block $B_{2,4}$ aggregates two demands: $1 \to 4$ and $5 \to 4$. In general, it might be beneficial for the block with null-volume demand to collect dispersive demands with low intensity.

However, it cannot be optimal to arrange a direct block for a null-volume demand that cannot transport any other demands. Null-volume demand of this type will be eliminated by a recursive procedure. For each subtree, we recursively remove the leaf o for which $f_{o,d} = 0$ and the edge connected to this leaf. Meanwhile, its neighbor node becomes a new leaf. Please see Appendix F for more details. By doing so, fewer nodes are considered in the subtree, and therefore fewer variables are included in the restricted model. Taking Figure 4(a) for example, assume that $f_{8,12} = 0$. In that case, the node can be removed from the subtree, reducing the number of variables by $8 = 2^{|\overline{P}_{8,12}|}$. On the other hand, if the demand from node 15 to 12 would be 0, the block variable $B_{15,12}$ cannot be removed when either $f_{19,12} > 0$ or $f_{16,12} > 0$.

6 Computational results

In this section, we first examine the service network design model for the one-block train formation problem using a small case from Lin et al. (2012). Then, based on a realistic Chinese marshaling railroad network, the parameters of the tree-based decomposition algorithm are tuned. Finally, the algorithm is tested on a bigger network to assess its performance on large scale instances. The model and algorithm are coded in MATLAB 2013a. ILOG CPLEX 12.6 is used as the underlying optimization solver. The computational experiments are conducted on a Dell Precision T7610 workstation with an Intel Quad Core E5 processor and 64 GB of RAM.

6.1 Small-scale instance

The small-scale instance from Lin et al. (2012) contains 19 yards and 23 links and is located in the Northeast of China. Firstly, we give a brief illustration of the input data and the preprocessing. Then, we present the output, including the optimal solution, and compare the optimal solution to the one found by the tree-based decomposition heuristic. Finally, we investigate the effect of the unitary rule and intree requirement on the solution.

6.1.1 Data preparation and preprocessing

Parameters of the physical network and demands are listed in Lin et al. (2012). Assume that all train sizes take $m_{i,j} = 50$ cars, $\forall i, j \in V$; each track could park $L = 200$ cars in every station, and the reserve ratios of classification capacity and sort tracks are $\phi_i = 15\%$, $\varphi_i = 0\%$, $\forall i \in V$, respectively. In Table 8, we present the distance of all links, obtained from the website Huochepiao (2017).

Table 8: Distance of links for the small-scale instance (unit: km)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1		268																	
2	268		411					236											
3		411		104							93								
4			104		184														
5				184		355							131						
6					355		168										110		
7						168												163	
8		236							212										
9								212		170									
10									170		180								
11			93							180		168							
12											168		208						
13					131							208		229					
14													229		141				
15														141		148			80
16															148		86		204
17						110										86		84	
18							163										84		
19															80	204			

As explained in §4.1, for a given origin and destination, all valid classification stations must be located on the given path for each CBA variable. In this paper, we use Floyd's algorithm to generate shortest paths for all demands in advance. According to formula (1), the set $\Omega_{i,j}$ should be filled with all combinations choosing exactly k elements from $\overline{P}_{i,j}$, $k = 1, 2, \cdots, |\overline{P}_{i,j}|$, $\forall i, j \in V$. Lin et al. (2012) do not take the consumption of classification capacity at the demands' destinations into consideration. Therefore, in order to allow to compare our results to theirs, constraint (11) is adjusted as follows

$$\sum_{o,d \in V} \sum_{\{I_{o,d}^{n_1,n_2,\cdots,n_l} \in \Omega_{o,d} | j \in \{n_1,\cdots,n_l\}\}} f_{o,d} x_{o,d}^{n_1,n_2,\cdots,n_l} \leq \alpha_j (1 - \phi_j), \forall j \in V. \tag{22}$$

6.1.2 Optimal solution

The exact model (3), (4), (9), (10), (12), (13), (22) contains 5,106 variables and 9,908 constraints and computes the optimal solution in less than 1 second. The optimal value is 45,421 car*hour, where the total train cost and classification cost are 38,375 car*hour and 7,046 car*hour, respectively. For this solution, the capacity workload (number of cars classified), the percentage of capacity utilized, the track usage and the percentage of tracks occupied in each yard are shown in Table 9.

In the optimal solution, there are 69 one-block train services. In more detail, 46 blocks are provided between adjacent stations, whereas 23 direct trains connect non-adjacent stations without stopping at the intermediate stations. According to formula (2), the total number of cars contained in each block after consolidation is listed in Table 10 for adjacent blocks and in Table 11 for the

Table 9: Capacity and track usage in each station

Station	Capacity		Track	
	workload (car)	percentage (%)	usage	percentage (%)
1	1028.51	73.99	2.50	62.49
2	805.90	86.12	2.87	71.65
3	1715.13	95.91	8.96	89.58
4	787.60	90.99	8.04	89.30
5	2312.63	95.17	18.56	88.39
6	277.99	37.00	1.05	17.53
7	301.32	56.83	1.51	50.23
8	98.12	32.12	0.49	24.60
9	287.05	55.87	1.44	47.86
10	289.33	58.16	1.23	30.82
11	233.28	35.60	1.17	38.89
12	122.68	32.97	0.62	20.51
13	310.80	59.61	2.47	49.42
14	177.97	51.06	2.54	50.74
15	968.23	83.28	3.58	71.59
16	861.18	85.94	1.51	75.28
17	254.01	62.17	1.32	44.01
18	968.02	88.71	1.70	34.04
19	825.67	80.87	1.59	26.46
average	664.50	76.10	3.32	61.29

remaining blocks. These two types of blocks have significantly distinctive means: 92.60 cars and 363.72 cars, respectively. Compared with an average intensity of 31.61 cars for all demands, the effectiveness of consolidation is quite apparent.

As we explained in §1, the intree constraint is memoryless and leads to predictable policies. Based on all CBA variables in the optimal solution, the consolidation strategy is given in Table 12. For the cell located at the i-th row and j-th column, the element k means that demand from i to j will be firstly classified at station k when $k \neq j$, otherwise there is a direct train from i to j. When implementing this consolidation strategy, the full itinerary for each demand can be recovered recursively. For example, OD $7 \to 2$ will be grouped into block $B_{7,5}$ and firstly classified at station 5. Next it will be transferred to another block $B_{5,3}$ and reclassified at station 3. Then, a direct train $3 \to 2$ would carry it to its final destination. In other words, OD $7 \to 2$ adopts the CBA plan $I_{7,2}^{5,3}$.

6.1.3 Comparison with the literature

Based on the small-scale instance from Lin et al. (2012), we apply the tree-based decomposition algorithm and compare the approaches with respect to their performance and solution feasibility.

The tree-based decomposition is configured using the settings obtained in §6.2.2 and determines the feasible solution that is shown in Table 13. Comparing this table to Table 12, there are some additional empty cells. These correspond to null-volume ODs and were deliberately eliminated (please see the discussion in §5.3.2). However, we cannot ignore all null-volume demands in the model. If we would do so, this would lead to infeasibility for other demands. For example, $P_{10,14} = \{10, 11, 12, 13, 14\}$. If the decision for $11 \to 14$ (with volume 0) is not included, the demand $10 \to 14$

Table 10: Adjacent block

Origin	Destination	Volume (car)	Origin	Destination	Volume (car)
1	2	5.95	11	3	161.93
2	1	21.96	11	10	22.02
2	3	238.65	11	12	49.39
2	8	94.62	12	11	54.47
3	2	453.76	12	13	68.61
3	4	379.81	13	5	162.23
3	11	110.92	13	12	73.29
4	3	221.89	13	14	49.76
4	5	245.53	14	13	100.69
5	4	189.79	14	15	33.50
5	6	85.69	15	14	128.21
5	13	141.50	15	16	35.52
6	5	135.62	15	19	30.98
6	7	32.17	16	15	46.04
6	17	42.61	16	17	128.62
7	6	11.13	16	19	34.49
7	18	20.18	17	6	181.17
8	2	95.09	17	16	60.16
8	9	3.29	17	18	22.70
9	8	3.50	18	7	2.88
9	10	45.95	18	17	82.78
10	9	23.42	19	15	35.08
10	11	67.89	19	16	24.31

Table 11: Non-adjacent block

Origin	Destination	Volume (car)
1	3	494.00
2	4	218.00
3	1	365.45
3	9	260.34
3	10	221.36
4	1	641.10
4	2	251.10
4	19	247.80
5	3	205.70
5	7	266.27
5	15	644.65
5	16	741.19
5	18	925.14
5	19	512.40
7	5	270.05
9	3	237.71
10	3	155.25
13	15	208.96
14	5	373.18
15	5	521.23
16	5	91.96
18	5	254.71
19	5	258.12

Table 12: Consolidation strategy of CPLEX

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	–	2	3	3	3	2	3	2	2	2	3	3	2	2	3	3	3	2	3
2	1	–	3	4	4	4	4	8	8	8	3	3	4	4	4	4	4	4	4
3	1	2	–	4	4	4	4	2	9	10	11	11	4	4	4	4	4	4	4
4	1	2	3	–	5	5	5	2	3	3	3	3	5	5	5	5	5	5	19
5	3	3	3	4	–	6	7	4	3	3	3	13	13	13	15	16	6	18	19
6	5	5	5	5	5	–	7	5	5	5	5	5	5	17	17	17	17	17	17
7	5	5	5	5	5	6	–	5	5	5	5	5	5	18	18	18	18	18	18
8	2	2	2	2	2	2	2	–	9	9	9	9	9	9	9	9	2	2	9
9	8	8	3	3	3	3	3	8	–	10	10	10	10	10	10	10	10	3	10
10	9	9	3	3	3	3	3	9	9	–	11	11	11	11	11	11	11	3	11
11	3	3	3	3	3	3	3	10	10	10	–	12	12	12	12	12	3	3	12
12	11	11	11	11	13	13	13	11	11	11	11	–	13	13	13	13	13	13	13
13	5	5	5	5	5	5	5	12	12	12	12	12	–	14	15	15	5	5	15
14	5	5	5	5	5	15	15	13	13	13	13	13	13	–	15	15	15	15	15
15	5	5	5	5	5	16	16	14	14	14	14	14	14	14	–	16	16	16	19
16	5	5	5	5	5	17	17	15	15	15	15	15	15	15	15	–	17	17	19
17	6	6	6	6	6	6	18	6	6	6	6	6	6	16	16	16	–	18	16
18	5	5	5	5	5	17	7	5	5	5	5	5	5	17	17	17	17	–	17
19	5	5	5	5	5	16	16	15	15	15	15	15	15	15	15	16	16	16	–

(with volume 0.03) cannot be sent to its destination after being firstly classified at 11.

In our study, car-to-block assignment variables are path-based and give rise to a linear model. In contrast, the arc-based approach in Lin et al. (2012) leads to a non-linear model. Table 14 compares

Table 13: Consolidation strategy of the tree-based decomposition algorithm

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
1	–	2	3	3	2	2	2	2	2	2	3	3	2	2	2	2	2	2	2
2	1	–	3	4	5	5	5	8	8	8	3	3	5	5	5	5	5	5	5
3	1	2	–	4	4	4	4	2	9	10	11	11	4	4	4	4	4	4	4
4	1	2	3	–	5	5	5	2	3	3	3	3	5	5	5	5	5	5	19
5	4	4	4	4	–	6	7	4	4	4	4	13	13	13	15	16	6	18	19
6	5	5	5	5	5	–	7	5	5	5	5	5	5	17	17	17	17	17	17
7	–	5	5	5	5	6	–	–	5	5	5	5	5	18	18	18	18	18	18
8	2	2	2	2	2	2	–	–	9	9	9	9	9	9	9	–	–	–	–
9	8	8	3	3	3	3	–	8	–	10	10	10	10	10	10	10	3	3	10
10	9	9	3	3	3	3	–	9	9	–	11	11	11	11	11	11	3	3	11
11	3	3	3	3	3	3	3	10	10	10	–	12	12	12	12	12	3	3	12
12	11	11	11	11	13	13	13	11	11	11	11	–	13	13	13	13	13	13	13
13	5	5	5	5	5	5	5	12	12	12	12	12	–	14	15	15	5	5	15
14	5	5	5	5	5	15	15	13	13	13	13	13	13	–	15	15	15	15	15
15	5	5	5	5	5	16	16	14	14	14	14	14	14	14	–	16	16	16	19
16	5	5	5	5	5	17	17	–	–	15	15	15	15	15	15	–	17	17	19
17	6	6	6	6	6	6	18	–	6	6	6	6	6	16	16	16	–	18	16
18	5	5	5	5	5	17	7	–	5	5	5	5	5	17	17	17	17	–	17
19	5	5	5	5	5	16	16	15	15	15	15	15	15	15	15	16	16	16	–

our results to that of Lin et al. (2012). According to the paper by Lin et al. (2012), Lingo did not find an optimal solution for the network with 19 stations using a non-linear model. In contrast, CPLEX can optimally solve our linear model for medium instances with 40 stations within 5 seconds. To conclude, our linear model is superior to the non-linear one with respect to both computation time and the ability to prove optimality.

Table 14: Comparison with the literature

	Lin et al. (2012)		This paper	
	Simulated annealing	Lingo	Tree-based decomposition	CPLEX
CPU time (second)	34	666	3	< 1
Gap (%)	0	0.74	0.15	0
Number of blocks	69	71	69	69
Total train cost (car*hour)	38,375	39,525	38,310	38,375
Total classification volume (car)	1,814	1,611	1,839	1,813
Total classification cost (car*hour)	7,046	6,237	7,181	7,046
Total cost (car*hour)	45,421	45,762	45,491	45,421

6.1.4 Unitary and intree rule relaxation

We now investigate three kinds of relaxations. First, we allow the car-to-block assignment variables to take real numbers instead of only binary values, i.e., we relax the unitary rule. In this case, the model is a linear mixed integer program. Second, removing constraints (8) means that the intree rule is relaxed. In the third relaxation, both rules are discarded. In Table 15, all three relaxed models have a little increase of classification cost but a larger decrease in direct train cost. This results in

lower total cost. When both rules are relaxed, the lowest cost are obtained. So, with less restrictions, a more adequate balance of resource utilization is obtained. Note that we increased the values of ϕ_i and φ_i for all $i \in V$ when computing these results.

Table 15: Chinese rules relaxation ($\phi_i = 50\%$, $\varphi_i = 50\%$, $\eta_{i,j} = 3$)

	Original	Relaxation		
		Intree	Unitary	Both two
Number of blocks	80	79	70	68
Total train cost (car*hour)	44,455	43,860	41,037	37,186
Total classification volume (car)	12,048	12,133	12,461	12,445
Total classification cost (car*hour)	4,852	5,247	6,575	6,541
Total cost (car*hour)	49,307	49,107	47,612	43,727

6.2 Medium-scale instance

Based on the Chinese railroad system, a marshaling network is constructed to tune the parameters and examine the effects of the enforcements of the tree-based decomposition algorithm.

6.2.1 Network and data

The Chinese railroad system is one of the largest railway networks in the world which consists of about 5200 stations, 120,000 kilometers of track, 20,000 locomotives and 716,000 railcars in 2014 (National Bureau of Statistics of China 2015). We build a skeleton network with 40 marshaling stations and 61 links. Parameters for each station are provided by China Railway Corporation (2017). The lengths of the links in the network are inquired from the website Huochepiao (2017).

The instance might be infeasible when given an arbitrary combination of demands. Based on the discussion in §5.2.3, all demands will consume the classification capacity of their destinations and occupy the sort tracks of their origination, no matter whether a direct block or any CBA plan is selected. The following inequality

$$\sum_{i \in V} f_{i,j} \leq \alpha_j (1 - \phi_j), \forall j \in V \tag{23}$$

can easily be deduced from formula (11). Similarly, according to formula (12), we must have

$$\sum_{j \in V} f_{i,j} \leq L \beta_i (1 - \varphi_i), \forall i \in V. \tag{24}$$

Therefore, demands are generated randomly while adhering to formulas (23) and (24) to ensure the instance is feasible in this study.

By convention, the quality of a solution for minimization problems is evaluated by the formula below

$$Gap = \frac{Z_{up} - Z_{low}}{Z_{up}} \times 100\%$$

where Z_{up} and Z_{low} are the upper bound and lower bound on the optimal value, respectively. We will specify how these bounds are computed when this measure is being used.

6.2.2 Parameter tuning

As mentioned in §5.1.2, the maximum number of nodes allowed in each subtree influences the result of the subtree decomposition. For the marshaling network, we generate 20 instances and set $K = 15, 20, 40$. The results of the subtree decomposition are summarized in Table 16. There will be more subtrees with smaller size when the node size parameter takes a smaller value.

Table 16: Characteristics of the tree-based decomposition

Average	15	20	40
# of nodes per subtree	5.48	6.23	7.18
# of subtrees	325.30	256.45	205.70

The feasible solution depends on the order in which all subtrees are solved. On the one hand, the number of variables has a direct impact on the size of the restricted model. On the other hand, the sum of the volumes of the demands involved in each subtree makes an immediate contribution to the objective function. Therefore, two metrics are proposed.

$$Var(T) = \sum_{o \in S} \sum_{1 \le i < j \le |P_{o,d}|} \frac{2^{j-i-1}}{\sum_{k \in S} \chi(P_{o,d}(i), P_{k,d}) \chi(P_{o,d}(j), P_{k,d})} \tag{25}$$

$$Vol(T) = \sum_{o \in S} \sum_{1 \le i < j \le |P_{o,d}|} \frac{f_{P_{o,d}(i), P_{o,d}(j)}}{\sum_{k \in S} \chi(P_{o,d}(i), P_{k,d}) \chi(P_{o,d}(j), P_{k,d})} \tag{26}$$

where d and S are the root node and leaves set of subtree T, respectively. $P_{o,d}(i)$ is the i-th station on path $P_{o,d}$. Indicator function $\chi(a, A)$ takes value 1 if $a \in A$, and value 0 otherwise. The aim of introducing it is to correct for double counting. If $\chi(P_{o,d}(i), P_{k,d}) = \chi(P_{o,d}(j), P_{k,d}) = 1$, the pair of nodes $P_{o,d}(i)$ and $P_{o,d}(j)$ connect a common section in both branches $\langle o, d \rangle$ and $\langle k, d \rangle$. The number of variables or the sum of the volumes of the subtree should be counted only once. Therefore, we divide each of these measures by the number of branches the common section is included in. Using the above 2 metrics to sort the subtrees in descending and ascending order yields 4 different orders in total.

A comparison of the gap and the runtime for 20 instances are illustrated in Figure 7 and Figure 8, respectively. Mean values are summarized in Table 17. The lower bound is the optimal value obtained by CPLEX. Var and Vol indicate that the order is determined according to formula (25) and (26), respectively. In general, using the total volume indicator almost is superior to the variable indicator in terms of gap. The larger the node size parameter, the shorter the runtime. The reason may be that there are more subtrees when this size parameter takes a smaller value. Of course, smaller subproblems are easier to solve. It is anticipated that a smaller node size parameter may be profitable for larger-scale networks, even though it results in a larger number of subproblems. Another

conclusion is that descending orders require more time than ascending orders. Apparently, there is a benefit when solving small subtrees first. As mentioned in §5.2.1, some variables of subtrees solved later may be fixed beforehand because they share a common section with subtrees solved earlier. The consequence is that the size of the subsequent restricted models may be smaller than they would otherwise be, so that the total time needed could be cut down. To balance the solution quality and the computation time, the tree-based decomposition parameters would be configured as follows: the volume indicator is used with an ascending order, and the node size parameter is set to the number of stations.

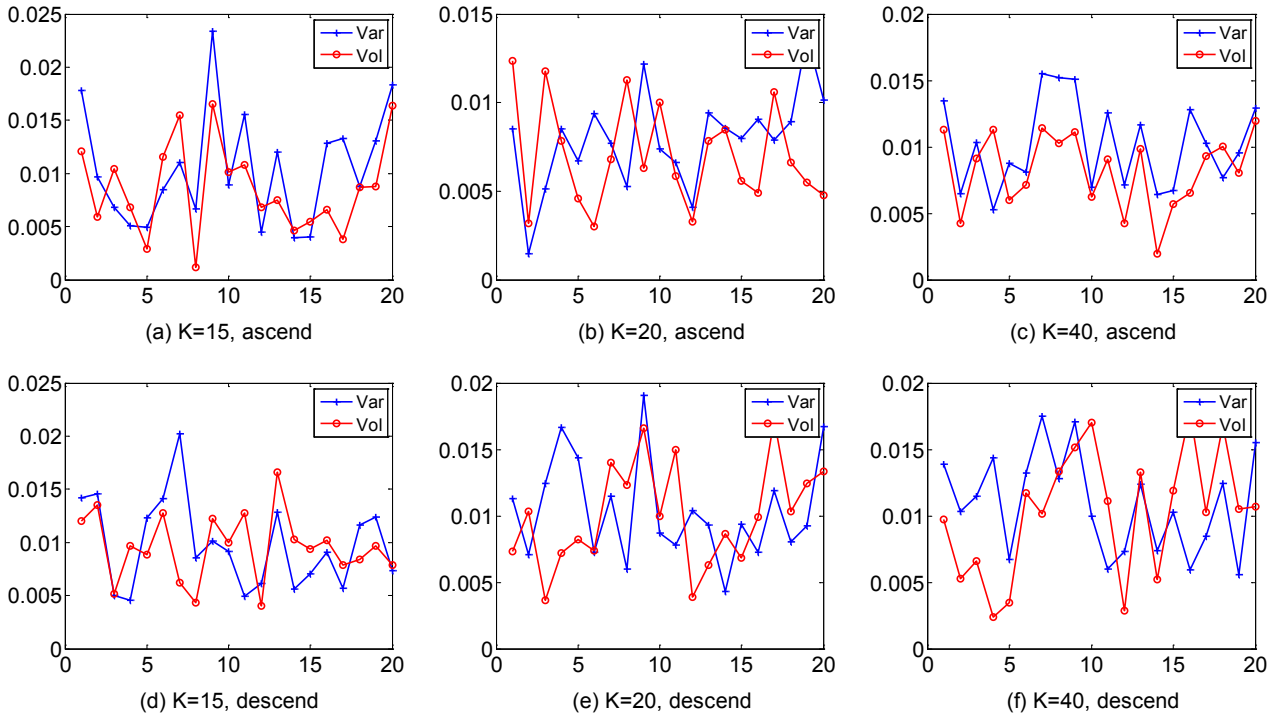


Figure 7: Comparison of Gap

Table 17: Comparison of four orders

		Ascending order			Descending order		
		15	20	40	15	20	40
Var	Gap (%)	1.05	0.79	1.01	0.98	1.04	1.09
	Runtime (second)	37.96	32.37	28.66	60.59	51.80	45.28
Vol	Gap (%)	0.86	0.70	0.82	0.96	1.00	1.03
	Runtime (second)	45.01	36.44	31.65	56.61	50.04	44.56

6.2.3 Enforcements effect

The effect of the greedy initialization and null-volume identification are examined in this section. Using the above configuration for the parameters, experiments are conducted with or without the enforcements. Figure 9 and Table 18 demonstrate that null-volume identification could save consid-

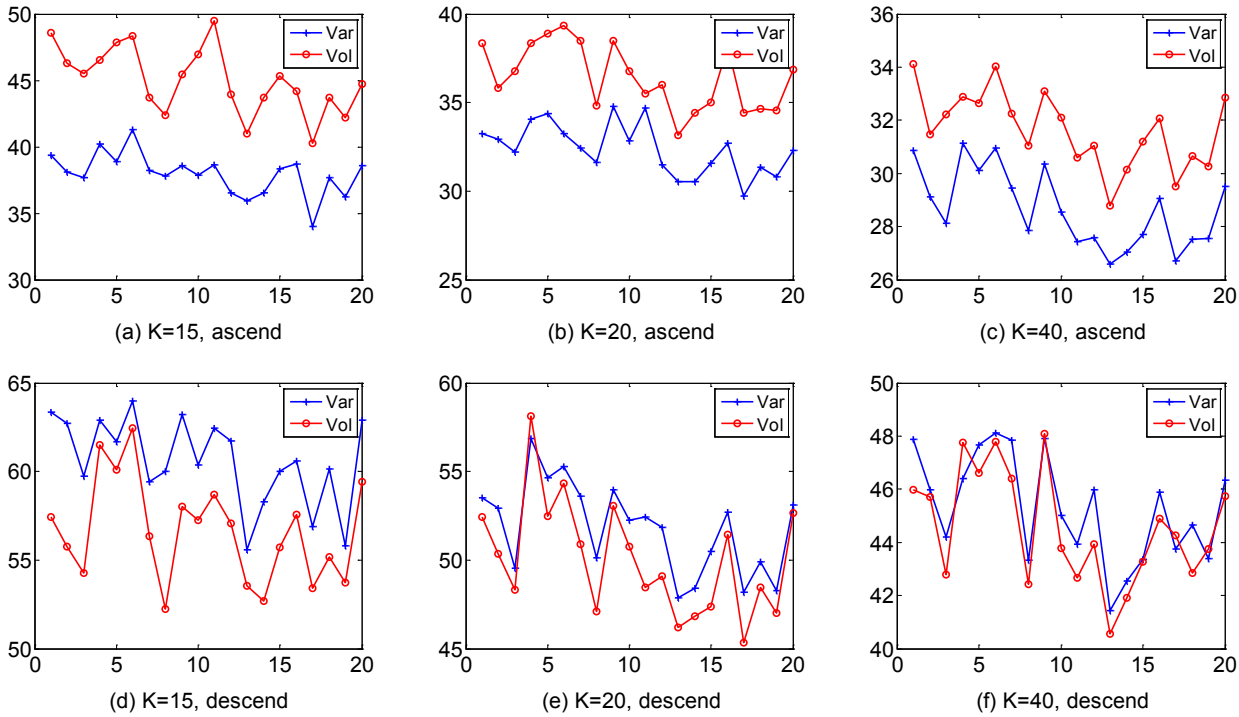


Figure 8: Comparison of runtime (second)

erable runtime. Both the gap and running time with and without the greedy initialization are very similar.

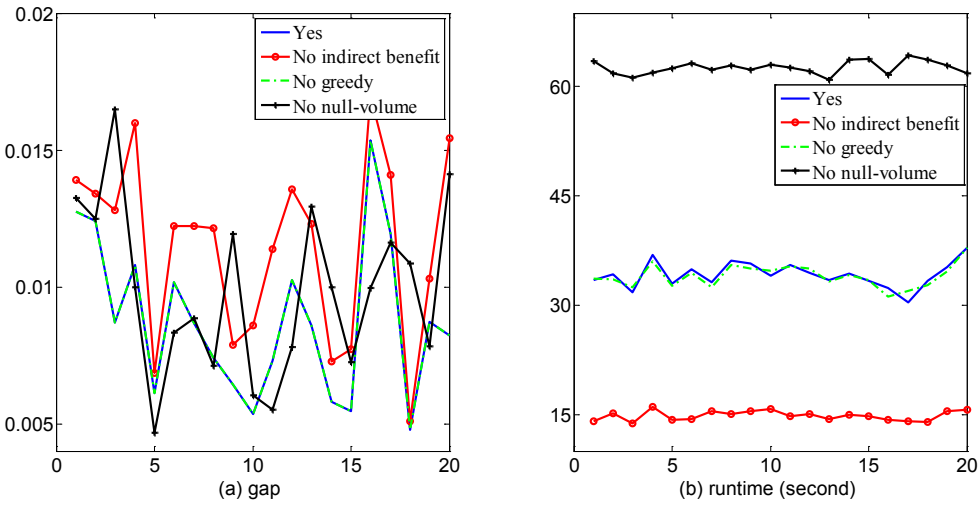


Figure 9: Comparison of enforcement effects

The benefit of substituting a block for a CBA plan is measured as the sum of the direct benefit and the indirect benefit. Evaluating the total benefit is complex, as can be seen in formula (14). In this formula, the complete solution must be scanned to determine the indirect benefit. This might take too much computation time. To deal with this, the benefit can simply be estimated only by the

Table 18: Comparison of enforcement effects

	Yes	No indirect benefit	No greedy	No null-volume identification
Gap (%)	0.88	1.15	0.88	0.99
Runtime (second)	32.10	14.35	31.74	58.67

direct benefit (corresponding to the first option in Figure 6), that is, by

$$c_{i,j}m_{i,j} - f_{i,j} \sum_{k\in\{n_1,n_2,\cdots,n_l\}} t_k,$$

when a block $B_{i,j}$ is substituted for the CBA plan $I_{i,j}^{n_1,n_2,\cdots,n_l}$. The results indicate a lower running time at the expense of a worse gap. To sum up, these experiments have clearly proven that the proposed enforcements perform well with respect to the tradeoff between effectiveness and efficiency.

6.3 Large-scale instance

In order to test the algorithmic performance on large-scale instances, a network with 158 links and 83 stations is considered. The network contains 40 marshaling stations and 43 stands at terminals or junctions of the rail system, see Figure 10. The data are acquired in the same way as in §6.2.

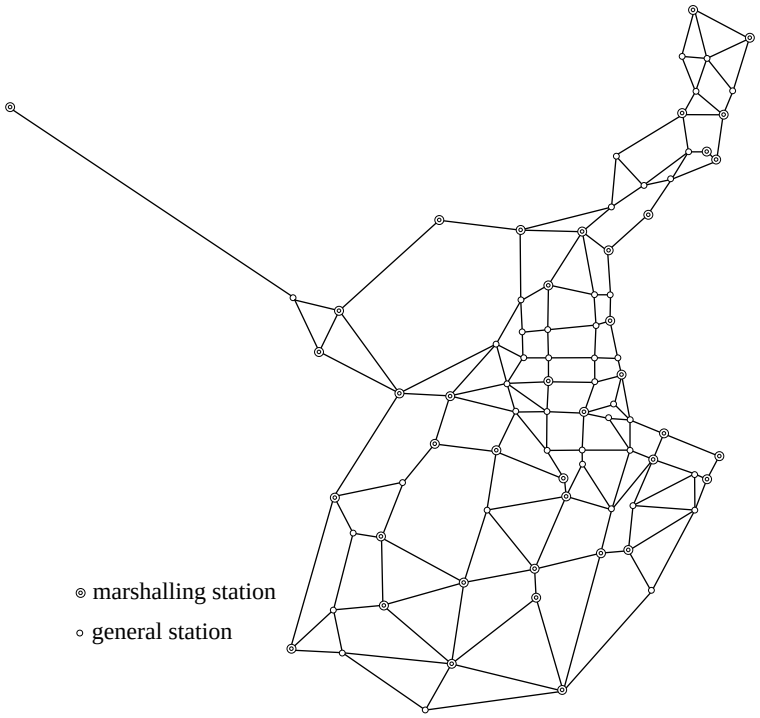


Figure 10: Realistic network of Chinese railroad system

Note that the coefficient matrices for the unequal and the equal constraints are sparse: The coefficient matrix has density $10^{-3}\%$. Therefore, we only record the non-zero cells' row coordinate, column coordinate and value. Table 19 lists the average model size over 5 instances.

Table 19: Model size

	Demands	Variables	Constraints	Non-zero elements in coefficient matrix
Average	5,689	7,323,976	14,641,312	149,999,290

We refer to it as a *cold start* when model (3)-(4), (9)-(13) is solved by CPLEX directly. In contrast, we call it a *warm start* when taking a known feasible solution as initialization. For the latter, the branch and bound procedure would be speeded up by pruning plenty of solutions with poor quality. Allowing a maximum runtime of 1 day, characteristics of the solutions for a cold start are given in Table 20. The mean gap is equal to 57.28%.

Table 20: Solutions for cold start (1 day)

Instance	Lower bound (car*hour)	Upper bound (car*hour)	Gap (%)
1	676,410	3,347,650	79.79
2	671,683	1,325,357	49.32
3	642,825	1,321,110	51.34
4	498,271	1,229,411	59.47
5	731,551	1,366,166	46.45
Average	644,148	1,717,939	57.28

Feasible results of the tree-based decomposition algorithm are presented in Table 21, where lower bounds are obtained from Table 20 and upper bounds are the total cost of the feasible solution, that is, the sum of all restricted models' objectives. The mean runtime is 2.88 hours and the mean gap is just 13.39%. This means that the gap is reduced by 43.89% on average compared to the solutions reported in Table 20.

Table 21: Solutions for tree-based decomposition algorithm

Instance	Runtime (hour)	Total cost (car*hour)	Gap (%)	Gap decrement (%)
1	1.44	773,002	12.50	67.30
2	2.78	777,629	13.62	35.70
3	3.86	741,876	13.35	37.99
4	2.49	572,230	12.92	46.55
5	3.82	855,963	14.53	31.92
Average	2.88	744,140	13.39	43.89

Again allowing a maximum runtime of 1 day, results for warm starts are given in Table 22. We can see that the lower bounds are exactly the same as those obtained with a cold start, and the upper bounds are exactly the same as those of the tree-based decomposition algorithm. That is to say, the warm start dose not make any improvement on the bounds.

For this network, more than a day-long is spent on generating input data (mainly on constructing the coefficient matrix of constraints) for the original model; and up to half an hour is needed just for loading the results into MATLAB. First of all, we observe that CPLEX cannot produce a reasonable

Table 22: Solutions for warm start (1 day)

Instance	Lower bound (car*hour)	Upper bound (car*hour)	Gap (%)
1	676,410	773,002	12.50
2	671,683	777,629	13.62
3	642,825	741,876	13.35
4	498,271	572,230	12.92
5	731,551	855,963	14.53
Average	644,148	744,140	13.39

solution within acceptable computation time. Furthermore, CPLEX cannot improve the solution quality of a given feasible solution easily, as demonstrated by the solutions obtained with a warm start. Thirdly, the state-of-the-art optimization solver cannot be confronted with instances of much larger scale without it running out of memory. In contrast, the tree-based decomposition algorithm is flexible and capable of deriving high quality solutions. It does so by subdividing the large-scale problem into a series of separable subproblems with smaller scale. By solving the subproblems sequentially, the storage requirements and computational effort are greatly reduced.

7 Conclusions

We have addressed the one-block train formation problem in the railway freight transportation industry. We have considered two decision subproblems: the block design that specifies which blocks are built in each yard; and the car-to-block assignment that details which railcars are allocated to which blocks. Based on the Chinese railway background, two specific rules are taken into account that are mostly ignored in other countries' railway reality and are also rarely encountered in scientific research. The unitary rule imposes each demand to be delivered on a single path and to be executed by a single classification itinerary. The intree rule requires that all railcars with the same destination must be operated in exactly the same way once they meet at an intermediate yard. In essence, the one-block TFP could be labeled as a tree-based service network design problem.

We contribute to the literature by proposing a linear binary programming formulation that seeks to minimize the sum of train cost and classification delay in the tactical level subject to the limitation on classification capacity and sort tracks at each station and the two specific requirements. Our exact model can be used to find provably optimal solutions for small and medium-scale networks.

For each destination separately, the intree rule does not only require car paths to form an undirected tree, but also specifies the solution of the TFP to form a directed tree. Therefore, the original problem can be viewed as the addition of several subproblems represented as separable trees rooted at each destination. A novel two-stage tree-based decomposition approach has been implemented to solve large-scale instances of this problem. In the first stage, each tree is decomposed into some independent subtrees according to the bifurcation at the root node. In the second stage, each independent subtree is further divided into manageable subtrees when necessary as specified by a node size parameter.

For each subtree, a restricted model is sequentially constructed to find an optimal subplan for

the local network. The global TFP for the holistic network could be obtained by integrating all local solutions. We apply a discriminative strategy, where block design variables are regarded as lazy variables and car distribution variables are regarded as active ones that could be replaced by the former. Three innovative manipulations on resources are developed: reserve in advance, release when free and adjust if needed. These manipulations coordinate the resource utilization to avoid the solution space being empty. In addition, two strengthening enforcements are proposed to accelerate the solution process: greedy initialization and elimination of null-volume demand.

The tree-based decomposition algorithm separates the large scale problem into smaller subproblems so that the storage requirements and computational effort are greatly reduced. The developed algorithm has been validated on realistic instances and has shown the capability of deriving high quality solutions (∼13% gap) within reasonable time (∼3 hours), strongly outperforming CPLEX (∼ 44% decrement of the gap obtained after one day).

Acknowledgment

The research was supported by the National Natural Science Foundation of China (No. 51505309, 61603318), the Fundamental Research Funds for the Central Universities (No. 2682016CX118). The authors warmly thank Dennis Huisman (Econometric Institute, Erasmus University Rotterdam) and colleagues for their valuable suggestions on preliminary versions of this paper. The first author gratefully acknowledges the support from China Scholarship Council during his research visit at Rotterdam School of Management, Erasmus University Rotterdam.

References

Ravindra K. Ahuja, Claudio B. Cunha, and Guevenc Sahin. Network models in railroad planning and scheduling. *INFORMS 2005 Tutorials in Operations Research*, pages 54–101, 2005.

Ravindra K. Ahuja, Krishna C. Jha, and Jian Liu. Solving real-life railroad blocking problems. *Interfaces*, 37 (5):404–419, 2007.

Arjang A. Assad. Modelling of rail networks: Toward a routing/makeup model. *Transportation Research Part B: Methodological*, 14(1-2):101–114, 1980.

Arjang A. Assad. Analysis of rail classification policies. *INFOR: Information Systems and Operational Research*, 21(4):293–314, 1983.

Cynthia Barnhart, Hong Jin, and Pamela H. Vance. Railroad blocking: A network design application. *Operations Research*, 48(4):603–614, 2000.

Cynthia Barnhart, Niranjan Krishnan, Daeki Kim, and Keith Ware. Network design for express shipment delivery. *Computational Optimization and Applications*, 21(3):239–262, 2002.

Lawrence D. Bodin, Bruce L. Golden, Allan D. Schuster, and William Romig. A model for the blocking of trains. *Transportation Research Part B: Methodological*, 14(1-2):115–120, 1980.

Nils Boysen, Malte Fliedner, Florian Jaehn, and Erwin Pesch. Shunting yard operations: Theoretical aspects and applications. *European Journal of Operational Research*, 220(1):1–14, 2012.

China Railway Corporation. statistics, 2017. URL http://www.china-railway.com.cn/en/Services/statistics/.

Jean-Francois Cordeau, Paolo Toth, and Daniele Vigo. A survey of optimization models for train routing and scheduling. *Transportation Science*, 32(4):380–404, 1998.

Teodor Gabriel Crainic. Service network design in freight transportation. *European Journal of Operational Research*, 122(2):272–288, 2000.

Teodor Gabriel Crainic, Jacques-A. Ferland, and Jean-Marc Rousseau. A tactical planning model for rail freight transportation. *Transportation Science*, 18(2):165–184, 1984.

Armin Fugenschuh, Henning Homfeld, and Hanno Schülldorf. Single-car routing in rail freight transport. *Transportation Science*, 49(1):130–148, 2015.

Bernard Gendron, Teodor Gabriel Crainic, and Antonio Frangioni. Multicommodity capacitated network design. In Brunilde Sanso and Patrick Soriano, editors, *Telecommunications Network Planning*, pages 1–19. Springer US, Boston, MA, 1999.

Michael Francis Gorman. An application of genetic and tabu searches to the freight railroad operating plan problem. *Annals of Operations Research*, 78:51–69, 1998.

Michael Francis Gorman, John-Paul Clarke, Amir Hossein Gharehgozli, Michael Hewitt, Rene de Koster, and Debjit Roy. State of the practice: A review of the application of OR/MS in freight transportation. *Interfaces*, 44(6):535–554, 2014.

Gianfranco Guastaroba, M. Grazia Speranza, and Daniele Vigo. Intermediate facilities in treight transportation planning: A survey. *Transportation Science*, 50(3):763–789, 2016.

Ali E. Haghani. Formulation and solution of a combined train routing and makeup, and empty car distribution model. *Transportation Research Part B: Methodological*, 23(6):433–452, 1989.

Christopher L. Huntley, Donald E. Brown, David E. Sappington, and Bernard P. Markowicz. Freight routing and scheduling at CSX transportation. *Interfaces*, 25(3):58–71, 1995.

Huochepiao. licheng, 2017. URL http://www.huochepiao.com/licheng/.

Phil Ireland, Rod Case, John Fallis, Carl Van Dyke, Jason Kuehn, and Marc Meketon. The Canadian Pacific Railway transforms operations by using models to develop its operating plans. *Interfaces*, 34(1):5–14, 2004.

Ahmad I. Jarrah, Ellis Johnson, and Lucas C. Neubert. Large-scale, less-than-truckload service network design. *Operations Research*, 57(3):609–625, 2009.

Krishna C. Jha, Ravindra K. Ahuja, and Guevenc Sahin. New approaches for solving the block-to-train assignment problem. *Networks*, 51(1):48–62, 2008.

Jiangang Jin, Jun Zhao, and Derhorng Lee. A column generation based approach for the train network design optimization problem. *Transportation Research Part E: Logistics and Transportation Review*, 50:1–17, 2013.

Mark H. Keaton. Designing optimal railroad operating plans: Lagrangian relaxation and heuristic approaches. *Transportation Research Part B: Methodological*, 23(6):415–431, 1989.

Mark H. Keaton. Designing railroad operating plans: A dual adjustment method for implementing Lagrangian relaxation. *Transportation Science*, 26(4):263–279, 1992.

Abdullah A. Khaled, Mingzhou Jin, David B. Clarke, and Mohammad A. Hoque. Train design and routing optimization for evaluating criticality of freight railroad infrastructures. *Transportation Research Part B: Methodological*, 71(Supplement C):71–84, 2015.

Ohkyoung Kwon, Carl D. Martland, and Joseph M. Sussman. Routing and scheduling temporal and heterogeneous freight car traffic on rail networks. *Transportation Research Part E: Logistics and Transportation Review*, 34(2):101–115, 1998.

Boliang Lin, Zhimei Wang, Lijun Ji, Yaming Tian, and Guoqing Zhou. Optimizing the freight train connection service network of a large-scale rail system. *Transportation Research Part B: Methodological*, 46(5):649–667, 2012.

Guglielmo Lulli, Ugo Pietropaoli, and Nicoletta Ricciardi. Service network design for freight railway transportation: The Italian case. *Journal of the Operational Research Society*, 62(12):2107–2119, 2011.

Angel Marin and Javier Salmeron. Tactical design of rail freight networks. part I: Exact and heuristic methods. *European Journal of Operational Research*, 90(1):26–44, 1996a.

Angel Marin and Javier Salmeron. Tactical design of rail freight networks. part II: Local search methods with statistical analysis. *European Journal of Operational Research*, 94(1):43–53, 1996b.

David R. Martinelli and Hualiang Teng. Optimization of railway operations using neural networks. *Transportation Research Part C: Emerging Technologies*, 4(1):33–49, 1996.

National Bureau of Statistics of China. National data, 2015. URL http://data.stats.gov.cn/easyquery.htm?cn=C01.

Harry N. Newton, Cynthia Barnhart, and Pamela H. Vance. Constructing railroad blocking plans to minimize handling costs. *Transportation Science*, 32(4):330–345, 1998.

Vedat Verter and Bahar Y. Kara. A path-based approach for hazmat transport network design. *Management Science*, 54(1):29–40, 2008.

Nicole Wieberneit. Service network design for freight transportation: A review. *OR Spectrum*, 30(1):77–112, 2008.

Masoud Yaghini, Amir Foroughi, and Behnam Nadjari. Solving railroad blocking problem using ant colony optimization algorithm. *Applied Mathematical Modelling*, 35(12):5579–5591, 2011.

Endong Zhu, Teodor Gabriel Crainic, and Michel Gendreau. Scheduled service network design for freight rail transportation. *Operations Research*, 62(2):383–400, 2014.

Appendices

A Identification of leaves

Input: Number of nodes N, root node d, paths $P_{i,d}$ for all $i \in \{1, 2, \cdots, N\} - \{d\}$
Output: Leaves set S

initial $i \leftarrow 1, S \leftarrow \{1, 2, \cdots, N\} - \{d\}$;
while $i \leqslant |S|$ **do**
    if $S \cap \overline{P}_{S(i),d} \neq \emptyset$ **then**
        $S \leftarrow S - \overline{P}_{S(i),d}$;
        $i \leftarrow 1$;
    else
        $i \leftarrow i + 1$;
    end if
end while

B Identification of independent subtrees

Input: Root node d, leaves set S, paths $P_{i,d}$ for all $i \in S$
Output: Independent subtrees set T

initial: $G \leftarrow$ neighbors of the root, $T_j \leftarrow \emptyset$, $j = 1, 2, \cdots, |G|$
for $i = 1$ to $|S|$ **do**
    for $j = 1$ to $|G|$ **do**
        if $G(j) \in \langle S(i), d \rangle$ **then**
            $T_j \leftarrow T_j \cup \{\langle S(i), d \rangle\}$;
            break;
        end if
    end for
end for

C Decomposition of independent subtrees

Input: Root node d, leaves set S, paths $P_{i,d}$ for all $i \in S$, node size parameter K, independent subtrees T_j with a number of nodes D_j, $j = 1, 2, \cdots, |G|$

Output: Subtrees set $\widetilde{T}$

initial: $m \leftarrow |G|$, $i \leftarrow 0$, $l \leftarrow 0$, $\widetilde{T} \leftarrow \emptyset$
while $i \leq m$ **do**
    $i \leftarrow i + 1$;
    if $D_i > K$ **then**
        $Q \leftarrow$ critical nodes of T_i;
        $T_j \leftarrow \emptyset, j = m+1, \cdots, m+|Q|$;
        for $k = 1$ to $|T_i|$ **do**
            for $j = 1$ to $|Q|$ **do**
                if $Q(j) \in T_i(k)$ **then**
                $T_{m+j} \leftarrow T_{m+j} \cup T_i(k)$;
                break;
                end if
            end for
        end for
        $D_j \leftarrow$ number of nodes of subtree $T_j, j = m+1, \cdots, m+|Q|$;
        $m \leftarrow m + |Q|$;
    else
        $l \leftarrow l + 1$;
        $\widetilde{T}_l \leftarrow T_i$;
    end if
end while

D A proposition for the bifurcation node

Proposition 2. *Let a rooted tree be decomposed into a set of mutually independent subtrees in such a way that the degree of the root is 1 in each subtree. If a subtree has only one branch, then it has no bifurcation nodes; otherwise it has a unique bifurcation node.*

Proof. The first case is trivial just by the definition of a branch. For the second case, we give a proof by contradiction. Assume that there would be more than two bifurcation nodes for a subtree rooted at d. Suppose i and j are two of them, i.e., both i and j are closest to the root with degree at least 3. We consider the intersection between the paths $P_{i,d}$ and $P_{j,d}$. Two branches in a subtree must have identical nodes except the root node, otherwise they would have been decomposed into independent subtrees. Therefore, the intersection between $P_{i,d}$ and $P_{j,d}$ contains at least one non-root node. Among the nodes in this intersection, the farthest one also is a bifurcation node and is closer to the root. This contradicts the assumption that i and j are closest. □

E Solution and resource adjustment

Input: Number of stations N, matrix H, cell W, available classification capacity γ_k, available sort track δ_k, $k = 1, 2, \cdots, N$

Output: H, W, γ_k, δ_k, $k = 1, 2, \cdots, N$

for $i = 1$ to $|W|$ **do**
  if $W\{i\} \neq \emptyset$ **then**
    $A \leftarrow H(i,1) \cup W\{i\} \cup H(i,2)$;
    $j \leftarrow 1$;
    while $j \leq |W\{i\}|$ **do**
      $k \leftarrow j + 2$;
      while $k \leq |W\{i\}| + 2$ **do**
        if there is a block from $A(j)$ to $A(k)$ **then**
          $\gamma_k \leftarrow \gamma_k + f_{H(i,1),H(i,2)}$, $k \in A(j+1 : k-1)$;
          $\delta_k \leftarrow \delta_k + f_{H(i,1),H(i,2)}$, $k \in A(j+1 : k-1)$;
          $W\{i\}(j : k-2) = [\,]$;
          $A \leftarrow H(i,1) \cup W\{i\} \cup H(i,2)$;
          $k \leftarrow j + 2$;
        else
          $k \leftarrow k + 1$;
        end if
      end while
      $j \leftarrow j + 1$;
    end while
  end if
end for

F Null-volume OD elimination

Input: Number of stations N, root node d, leaves set S, path $P_{i,d}$, volume $f_{i,d}$ for all $i \in S$
Output: Null-volume set R

initial: $R \leftarrow \emptyset$, $i \leftarrow 1$, $n \leftarrow 0$
while $i \leq |S|$ **do**
  if $f_{S(i),d} = 0$ **then**
    $n \leftarrow n + 1$;
    $R(n) \leftarrow [S(i), d]$;
    $U \leftarrow P_{S(i),d}$;
    $j \leftarrow 2$;
    while $\{j < |U|$ and $f_{U(j),d} = 0\}$ **do**
      $flag \leftarrow 0$;
      for $k = 1$ to $|S|$ **do**
        if $k \neq i$ and $U(j) \in P_{S(k),d}$ **then**
          $flag \leftarrow 1$;
          break;
        end if
      end for
      if $flag = 0$ **then**
        $S(i) \leftarrow U(j)$;
        $n \leftarrow n + 1$;
        $R(n) \leftarrow [U(j), d]$;
      else
        $S(i) \leftarrow [\,]$;
        break;
      end if
      $j \leftarrow j + 1$;
    end while
  end if
  $i \leftarrow i + 1$;
end while