Mixed Integer Linear Programming and Solution for the Multi-tank Train Stowage Planning Problem of Coils in Steel Industry

Fei YANG,^{1,2)} Lixin TANG,^{2)*} Yuan YUAN²⁾ and Kun LI²⁾

1) State Key Laboratory of Synthetical Automation for Process Industries, Northeastern University, Shenyang, 110819, P. R. China.

2) The Logistics Institute, Northeastern University, Shenyang, 110819, P. R. China.

(Received on April 2, 2013; accepted on October 18, 2013)

We consider a multi-tank train stowage planning problem of coil (TSPP) where the given coils are assigned into the tanks of the train. Different from stowage problems in previous studies, this problem has its characteristics such as the multi-tank transportation mode, the minimum load restriction of each tank and the rigorous balance constraints. We formulate the problem as a novel mixed integer linear programming model which objectives are to maximize the total weight of loaded coils and to minimize the differences of stack positions of the coils in each tank. The NP-hardness of the problem and intractability of optimally solving the model motivate us to develop an improved tabu search algorithm to solve it approximately. The algorithm is initiated by a two-stage heuristic where a multi-exchange neighborhood is designed to further improve the initial solutions. And then K -cycle move is used regarded as a diversification strategy in tabu search. Computational results using real data from a specialty steel manufacturer show that for small problems, which can be solved optimally by the model, the proposed algorithm can generate close-to-optimal solutions. For large practical problems the algorithm can obtain good solutions within a shorter time compared with the upper bound.

KEY WORDS: steel industry; train logistics; coil stowage planning; tabu search algorithm.

1. Introduction

For steel industry, product cost mainly consists of raw material cost, manufacturing cost and logistics cost. Due to the increase of raw material price and the difficulties on reducing manufacturing cost, how to reduce the logistics cost becomes a main way to enhance enterprise competitiveness. The effective logistics operation management is a good way to reduce the logistics cost. The logistics operations include transportation, storage and material handling. Reasonable transportation can improve the customer satisfaction effectively and reduce the transportation logistics cost greatly. Generally speaking, there are three primary transportation modes for delivering steel products in steel industry: ship transportation which is suitable for long-distance transport, train transportation mainly for domestic transport, and truck transportation utilized as a short-haul transport. For an inland steel company, the train transportation plays an important role in logistics operations. For example, the steel plant under our consideration has the annual production throughput of about 300 million tons and more than 20 percent of the steel products (steel coils) are transported by train.

The train transportation of the steel products is as shown in **Fig. 1**. The stability of each tank of the train has to be maintained. To keep stability of each tank of the train and

to reduce the numbers of crane operation, an effective train stowage plan is critical.

For each arriving train, the steel company needs to make a loading plan. The plan is made in two phases. The first phase is to select steel coils to be loaded onto the train considering the destinations of the train, order priorities, quantities and due dates of customer orders and the positions of the coils in storage yard, called the train consolidation planning problem (TCPP), and these selected coils are called the

(a)

(b)

Fig. 1. The train transportation of the steel coils.

* Corresponding author: E-mail: qhjytlx@mail.neu.edu.cn

DOI: <http://dx.doi.org/10.2355/isijinternational.54.634>

activated coils. The second phase is to decide that how many tanks should be loaded in and how many activated coils should be loaded in which positions of tanks, considering the constraints such as the balance of coils of each tank and the stowage requirement, called the train stowage planning problem (TSPP). In addition, not all the activated coils should be necessarily loaded into the tanks due to the strict balance constraints. In this paper, we study the second-phase problem, the train stowage planning problem (TSPP) arising in the logistics system of a steel company.

In current practice, a rough stowage plan is made manually based on the planner's experiences. Due to the large problem size and the complicated practical restrictions, it is difficult for the planners to solve it effectively. Such a manual plan often results in the need for a large number of shuffles at destination ports and lower loading rate. More effective stowage planning method is needed.

There has been little research on the train stowage planning problem for steel products. Our literature survey only found one previous study on the ship stowage planning problem of coil. Umeda *et al.*¹⁾ studied a ship stowage planning problem of steel products including coils in 2001. The problem was considered as a three-dimensional allocation problem and solved using simulated annealing. In determining the positions of the stowed products, the weight balance of the ship, the loading ratio and the work efficiency of stowing were considered. The ship considered in the study had fixed slots for stowing the products. In our paper, the coils are loaded on only one layer into several tanks of the train, and the exact positions of the coils are determined by the weight and number of loaded coils, and the stowage requirements and the balance constraints are more rigorous.

Almost all the researches of stowage planning problem focus on container ship and we haven't found any study of coil stowage planning problem for train yet. For example, Avriel *et al.*²⁾ studied a stowage planning problem to minimize the number of container shifts needed for unloading them at destination port in 1998. Some constraints such as stability of the ship were not considered. Ambrosino *et al.*³⁾ addressed a stowage planning problem considering constraints such as container dimensions, weight and destinations as well as the balance of the ship in 2004. The objective was to minimize the total stowage time according to the positions of stowed containers on board. The problem was formulated as a binary integer programming model and solved using a three stage approach. Through preprocessing and prestowage in the first stages, the search space was reduced. By further relaxing some constraints the reduced model was then solved in the final stage. In 2006 Imai *et al.*⁴⁾ determined the container stowage and loading plans of a ship considering stacking configurations of containers in yard to maximize the ship stability and minimize the number of container shuffles during the loading process. A genetic algorithm was proposed to solve the problem and to obtain a set of non-dominated solutions. Álvarez⁵⁾ proposed an approach using the tabu search and multi-start techniques to generate vessel loading plans in reach-stacker based terminals, considering the vessel stowage requirements and the container stacking information in the yard as inputs in 2006.

There are significant differences between the stowage planning of coils and those of containers as follows.

1) The configuration of transporter is different between train and containership. The transporter of train includes many tanks which could be added or reduced flexibly and that of containership is a whole big cabin including upper deck and lower deck.

2) Containers are of standard sizes and the containers of the same size are stacked one on top of another on the ship. Steel coils, on the other hand, have different weight, diameters and widths. Due to the cylindrical shape, the coils cannot be stowed in the same way as containers.

3) In TSPP, the constraints for balance of tanks are more rigorous. For example, in a tank, the weight difference between left group and right group must be not more than 1 ton.

4) For each tank of a train, the positions for loaded coils are allocated according to the number of loaded coils for this tank, which is distinctly different from the fixed position of containership.

5) TSPP allows one or more coils to be left, that is to say, TSPP is a selective problem for loading coils whereas all of the containers must be loaded into the containership.

6) Underload is not allowed for train which is different from containership. There exists a lower limit of loaded weight for each tank in order to economize transportation cost whereas the containership must take the satisfaction of customers into consideration primarily.

More recently, Tang *et al.*⁶⁾ studied the item shuffling problems arising in the logistics system of steel production in 2012. An item shuffling problem in their paper was to optimize shuffling operations needed in retrieving a sequence of steel items from a warehouse served by crane. There were two types of such problems, plate shuffling problem (PSP) and coil shuffling problem (CSP), considering the item shapes. The problem PSP was modeled as a container storage location assignment problem according to reducing the shuffling numbers. In our paper, the shuffling is also considered by selecting coils with the same stack number into a tank.

The rest of this paper is organized as follows. In the next section the stowage planning problem is described in detail. In section 3 we formulated the TSPP as a mixed integer linear programming (MILP) model to maximize the objectives by considering some practical issues. Due to the complex loading constraints, an improved tabu search (TS) algorithm is developed in Section 4 to obtain a good stowage plan, with the initial solution obtained by a two-stage heuristic. Computational results are shown in Section 5 to demonstrate the effectiveness of the TS algorithm. Section 6 concludes the paper.

2. Problem Description

Given a set of steel coils to be loaded onto a train, the problem is to determine the number of the loading tanks of the train and the location of each coil on the train, considering the sizes of the coils, the stability of the train and the convenience of crane operation. Not all the activated coils should be loaded onto the train according to the strict loading requirement.

For the train transportation, the stack yard is just like a production buffer for the former process. **Figure 2** indicates

Fig. 2. The sketch map of train transportation stack yard.

Fig. 3. The turning state of the loaded coils.

the sketch map of the train stack yard and the operation area.

The train loading depot is shown in Fig. 2. It is not an open yard for stacking hot rolling coils. The symmetrical stack includes two same sections *A* and *B*, in column *A1* to *AE* about 12-17 coils can be laid based on the dimension of the coils and 6-8 coils in column *AF* to *AH*. The tanks cannot be towed out of yard by loco only if the loading weight of each tank is larger than a limitative quantity. There are two traveling cranes for operating the import and export coils. And a coil upender is used for turning coils when stowed into the tank such as Fig. 3. *A* denotes the state of the coil in stack yard, and *B* denotes the state of the coil in the tank.

In train stowage planning problem of coil (TSPP), the loading requirements are strict. In order to satisfy the requirements of railway administration, the loading constraints are required as follows:

- 1) The total weight of loading coils is not more than a maximal value.
- 2) There is a limitative weight of the coil loaded in the middle position of each tank.
- 3) The weight difference between left group and right group is not more than a fixed value as shown in Fig. 4.
- 4) The weight difference between the coils of the same line in the vertical direction of the tank should be less than a maximal value. For example, in case (c) of Fig. 4, the weight differences between the upper and the lower coils, such as coil 1 and coil 2, coil 5 and coil 6, are not more than a maximal value respectively, in case (d), the weight difference between coil 1 and coil 2, coil 3 and coil 4, coil 5 and coil 6, coil 7 and coil 8 are not more than the same maximal value respectively.

Two objectives are considered in TSPP.

- 1) Maximize the loading of the train

According to the loading requirement above, the more the coils are loaded into the train the better the stowage plan is. Not all the activated coils need to be loaded onto the train, but underload is not allowed for the train. There exists a limit of loaded weight for each tank in order to economize

Fig. 4. Loading positions of different number of coils.

Table 1. An example of eight coils with their stack position.

Tank	Coil number	Stack position	Tank	Coil number	Stack position
1	1	AD03	2	5	B305
1	2	AD04	2	6	B505
1	3	AD05	2	7	B501
1	4	AD06	2	8	B301

transportation cost.

- 2) Minimize the transportation distances of cranes for loading coils

As the number of tanks can be preprocessed before the process of loading, the problem is to assign which coils to be loaded into the which tank. The coils are loaded in two layers in the stack yard, and in the same tank the loaded coils from the same stack may avoid shuffles. We regard the situation that the coils loaded in same tank have near stack positions as a better stowage planning. For example, in Table 1, eight coils will be loaded into the tanks. A better stowage planning considering the transportation of crane is that coil 1 to coil 4 are loaded into tank 1, and the other four coils are loaded into tank 2 (suppose it is satisfied with all loading constraints). The section is expressed as the first letter of the stack position, and the column is expressed by the second letter of the stack position. According to the layout of loading depot, we consider two kinds of punished costs for reducing the distance of crane in objective function. The first one is the times of section differences for loaded coils in a tank denoted by s_j^1 , j denotes tank number. If the coils loaded in tank j are from the same section, $s_j^1 = 0$, we can obtain $s_1^1 = 0$ and $s_2^1 = 0$ in the example. The second one is the times of section and column differences for loaded coils in a tank denoted by s_j^2 . It is defined by the anterior two letter of stack position. If the coils loaded in tank j are from the same section but not the same column, $s_j^2 = 1$. For coil 1 to coil 4, the anterior two letters of their stack positions are same, then we can get $s_1^2 = 0$, and the second letters of stack position of coil 5 is 3, which is different from that of

coil 6 and coil 7, then we can get the times of section and column differences of coil 5 is 2, the calculation for the other three coils in tank 2 can be obtained in the same way, then $s_2^2 = 2 + 2 + 2 + 2 = 8$.

The TSPP is to determine the locations of a given set of steel coils with the same destination in the tanks of the train, considering the coil information and the loading requirements, and to optimize a combination of the above objectives.

3. Model

The TSPP has some complex constraints and multiple objectives which are very different from the container stowage planning problem studied in the previous literatures, and a MILP model is proposed. The objective function is to maximize the total weight of loaded coils and minimize the total times of different section and column of each two coils in tanks. In order to express this model detailedly, the position information of the tank is defined as **Fig. 5**. Usually, the number of loading coils in each tank will not exceed 9.

Known parameters:

- C The set of activated coils, $C = \{1, 2, \dots, n\}$, where n is the number of coils.
- J The set of tanks, $J = \{1, 2, \dots, m\}$, where m is the number of tanks.
- W_i The weight of coil i , $i \in C$.
- K The set of positions of tank.
- S_j^1 The times of section differences for loaded coils in tank j , $j \in J$.
- S_j^2 The times of section and column differences for loaded coils in tank j , $j \in J$.
- T_j The loading capacity of tank j , $j \in J$.
- P The underload weight of the tank.
- B_1 The maximal counterbalance between the left and right coil groups.
- B_2 The maximal counterbalance between upper and lower coils in the same coil groups, it exists only if the number of loading coils in a tank is more than five.
- B_3 The maximal weight of the middle coil in tank, it exists only if the number of loading coils in a tank is odd number, and we suppose that the last position is for the middle coil in tank.
- z_i The coefficient of objective function, $i=1, 2, 3, 4$.
- $S_{i_1 i_2}$ The total times of section differences and section-column differences for coil i_1 and coil i_2 , $S_{i_1 i_2} = z_1 s_{i_1 i_2}^1 + z_2 s_{i_1 i_2}^2$, where

$$s_{i_1 i_2}^1 = \begin{cases} 1 & \text{if coil } i_1 \text{ and coil } i_2 \text{ are belong} \\ & \text{to a different section} \\ 0 & \text{otherwise} \end{cases}$$

$$s_{i_1 i_2}^2 = \begin{cases} 1 & \text{if coil } i_1 \text{ and coil } i_2 \text{ are the same section} \\ & \text{but in different column} \\ 0 & \text{otherwise} \end{cases}$$

Decision variables:

$$x_{ij}^k = \begin{cases} 1 & \text{if coil } i \text{ is assigned to position } k \text{ of tank } j \\ 0 & \text{otherwise} \end{cases}$$

Fig. 5. The position information of a tank.

$$y_{i_1 i_2 j} = \begin{cases} 1 & \text{if coils } i_1 \text{ and coils } i_2 \text{ are assigned to tank } j \\ 0 & \text{otherwise} \end{cases}$$

$$v_j = \begin{cases} 1 & \text{if tank } j \text{ is used} \\ 0 & \text{otherwise} \end{cases}$$

$$q_j = \begin{cases} 1 & \text{if the number of loaded coils is more than 5 in tank } j \\ 0 & \text{otherwise} \end{cases}$$

$$h_j = \begin{cases} 1 & \text{if the number of loaded coils is more than 7 in tank } j \\ 0 & \text{otherwise} \end{cases}$$

Integer programming model:

$$\text{Maximize } \sum_{i \in C} \sum_{j \in J} \sum_{k \in K} z_3 x_{ij}^k W_i - \sum_{j \in J} \sum_{i_1 \in C} \sum_{i_2 \in C} z_4 y_{i_1 i_2 j} S_{i_1 i_2} \dots (1)$$

Subject to

$$x_{ij}^k \in \{0, 1\} \quad \forall i \in C, \forall j \in J, \forall k \in K \dots (2)$$

$$\sum_{i \in C} x_{ij}^9 W_i \leq B_3 \quad \forall j \in J \dots (3)$$

$$v_j P \leq \sum_{i \in C} \sum_{k \in K} x_{ij}^k W_i \leq v_j T_j \quad \forall j \in J \dots (4)$$

$$\sum_{k=1}^4 \sum_{i \in C} x_{ij}^k W_i - \sum_{k=5}^8 \sum_{i \in C} x_{ij}^k W_i \leq B_1 \quad \forall j \in J \dots (5)$$

$$\sum_{k=1}^4 \sum_{i \in C} x_{ij}^k W_i - \sum_{k=5}^8 \sum_{i \in C} x_{ij}^k W_i \geq -B_1 \quad \forall j \in J \dots (6)$$

$$\sum_{i \in C} x_{ij}^1 W_i - \sum_{i \in C} x_{ij}^2 W_i \leq B_2 + (1 - q_j) M \quad \forall j \in J, M \text{ is a large natural number} \dots (7)$$

$$\sum_{i \in C} x_{ij}^1 W_i - \sum_{i \in C} x_{ij}^2 W_i \geq -B_2 - (1 - q_j) M \quad \forall j \in J, M \text{ is a large natural number} \dots (8)$$

$$\sum_{i \in C} x_{ij}^3 W_i - \sum_{i \in C} x_{ij}^4 W_i \leq B_2 + (1 - h_j) M \quad \forall j \in J, M \text{ is a large natural number} \dots (9)$$

$$\sum_{i \in C} x_{ij}^3 W_i - \sum_{i \in C} x_{ij}^4 W_i \geq -B_2 - (1 - h_j) M \quad \forall j \in J, M \text{ is a large natural number} \dots (10)$$

$$\sum_{i \in C} x_{ij}^5 W_i - \sum_{i \in C} x_{ij}^6 W_i \leq B_2 + (1 - q_j) M \quad \forall j \in J, M \text{ is a large natural number} \dots (11)$$

$$\sum_{i \in C} x_{ij}^5 W_i - \sum_{i \in C} x_{ij}^6 W_i \geq -B_2 - (1 - q_j) M \quad \forall j \in J, M \text{ is a large natural number} \dots (12)$$

$$\sum_{i \in C} x_{ij}^7 W_i - \sum_{i \in C} x_{ij}^8 W_i \leq B_2 + (1 - h_j) M \quad \dots\dots\dots (13)$$

$\forall j \in J, M$ is a large natural number

$$\sum_{i \in C} x_{ij}^7 W_i - \sum_{i \in C} x_{ij}^8 W_i \geq -B_2 - (1 - h_j) M \quad \dots\dots\dots (14)$$

$\forall j \in J, M$ is a large natural number

$$\sum_{i \in C} x_{ij}^k \leq 1 \quad \forall j \in J, \forall k \in K \quad \dots\dots\dots (15)$$

$$\sum_{j \in J} \sum_{k \in K} x_{ij}^k \leq 1 \quad \forall i \in C \quad \dots\dots\dots (16)$$

$$\left(\sum_{k \in K} x_{i_1 j}^k + \sum_{k \in K} x_{i_2 j}^k - 1 \right) / 2 \leq y_{i_1 i_2 j} \leq \left(\sum_{k \in K} x_{i_1 j}^k + \sum_{k \in K} x_{i_2 j}^k \right) / 2 \quad \dots\dots\dots (17)$$

$\forall j \in J, \forall i_1, i_2 \in C$

$$M(1 - q_j) \geq 6 - \sum_{i \in C} \sum_{k \in K} x_{ij}^k \quad \dots\dots\dots (18)$$

$\forall j \in J, M$ is a large natural number

$$M(1 - h_j) \geq 8 - \sum_{i \in C} \sum_{k \in K} x_{ij}^k \quad \dots\dots\dots (19)$$

$\forall j \in J, M$ is a large natural number

$$v_j \leq \sum_{i \in C} \sum_{k \in K} x_{ij}^k \quad \forall j \in J \quad \dots\dots\dots (20)$$

There are two terms in the objective function including the weight of loaded coils and the total times of different section and column of loaded coils in the same tank. Constraint (2) is the value range of x_{ij}^k . Constraint (3) is the loading requirement of middle coil. Constraint (4) ensures that the loading weight of each tank is satisfied with the requirement. Constraints (5) to (14) explain the restriction of the counterbalance in two coil groups and the counterbalance between the coils in same group. Constraints (15) and (16) are the assignment constraints, which ensure that each coil is stowed only in one location and each location has at most one coil. Constraints (17) to (20) are the relations of decision variables.

4. Improved Tabu Search Heuristic for TSPP

The NP-hardness of the problem and intractableness of optimally solving the model motivate us to develop an improved tabu search algorithm to solve it approximately. Nowadays, there are many meta-heuristic algorithms to act as combinational optimization methodologies. However, Tabu search was proposed by Glover⁷⁾ (1986) and by Hansen⁸⁾ (1986). And it has been proven to be one of the most effective local search techniques to find near-optimal solutions and can avoid being trapped at a local optimum for various scheduling problems in previous works such as Nowicki and Smutnicki⁹⁾ (1998), Barbarosoglu and Ozgur¹⁰⁾ (1999), and Watsona *et al.*¹¹⁾ (2006), Norman¹²⁾ (1999), Grabowski and Wodecki¹³⁾ (2004). The TSPP under our consideration is to determine which coils are loaded onto which locations on the train and is similar to those in scheduling problems, which motivate us to adopt TS to solve our problems. This algorithm is based on a neighborhood or local search technique and its principle is to make the recent

moves tabu to avoid falling into local optimum.

The main components of TS include initial solution, neighborhood, tabu list, stop criteria, speedup strategies. Each of the components is described as follows.

4.1. Initial Solution

This initial solution is a base solution for TS algorithm and the quality of final solution depends on it to a great extent. For the TSPP, to load more activated coils into the tanks is most important. According to the know-how of the planners in practice, we propose an effective two-stage heuristic to obtain an initial solution aiming for reducing the number of the coils left unloaded.

4.1.1. Two-stage Heuristic for Obtaining the Initial Solution

In heuristic, the obtained solution is a feasible solution and all the constraints must be satisfied. Simulating the operation process by the planners in practice, in the first stage of the heuristic the activated coils are assigned to tanks with the following rule. Suppose there are S stack of the activated coils, and the activated coils in each stack are sequenced in cross-order on the weight of coils. We can obtain sequences $C_s = \{C_1^s, C_{n^s}^s, C_2^s, C_{n^s-1}^s, \dots, C_{(n^s+1)/2}^s\}$, $s = 1, 2, \dots, S$, for n is an odd number, and $C_s = \{C_1^s, C_{n^s}^s, C_2^s, C_{n^s-1}^s, \dots, C_{n^s/2}^s\}$, for n is an even number, where C_1^s and $C_{n^s}^s$ mean the coil with maximum weight and the coil with minimum weight in stack s . Then we can obtain the final sequence of the activated coils $C = \{C_1, C_2, \dots, C_S\}$. M_j coils are selected from sequence C to tank j sequentially, and the sum of weights of these M_j coils must satisfy the constraint (4). We can get the initial value of M_j (it is the number of the assigned coils for tank $j, j \in J$). Then these coils are assigned to exact positions in tanks according to the following rule. In each tank the coils are loaded from left and right sides to the middle of the tank, then the coils are swapped in the same tank until the constraints (3), (5) to (14) are all satisfied. If any constraint is not satisfied, the corresponding value of M_j is set 0.

In the second stage of the heuristic, we need to reduce the number of the coils left unloaded, and three situations are discussed as follows. W_L is the total weight of the coils left unloaded, and the value of W_L is expected to be reduced considering the objective function of maximizing the weight of loaded coils.

(1) $W_L = 0$

It denotes that all coils have already been assigned to the tanks with their positions, and it is the current final initial solution for TS algorithm.

(2) $W_L \geq P$

If $W_L > T_j$, it means that the number of the coils left unloaded is enough to be loaded at least one tank, where $j \in L$ and L is the set of empty tank, $L \subset J$.

This situation may be caused by some constraints because of the improper sequence set C and we can use some adjustment steps as follows to solve it.

Step1 Swap the orders of two coils in set C_L , where C_L is the set of the coils left unloaded, $C_L \subset C$. According to the sequence set C_L , assign the coils left unloaded to the empty tank. Assign M_j coils of set C_L which subject to the constraint (4) to tank j and arrange the coils with their posi-

tions in tanks as the same way of the first stage. After fifty iterations, if $W_L \geq P$, then go to step 2, otherwise stops.

Step2 Swap two coils one of which belongs to set C_L and the other one is a member of set C_Y , where C_Y is the set of assigned coils, $C_Y \subset C$, $C_L \cap C_Y = \emptyset$, then judge whether the coils left unloaded can be assigned to the empty tanks with required constraints until $0 < W_L < P$ or $W_L = 0$.

After the adjustment of two steps, if $W_L = 0$, it means that all coils have already been assigned to the tanks and the current solution is the final initial solution for TS algorithm, $W_L \geq P$ implies some activated coils and empty tanks are left finally according to the constraints of the model.

(3) $0 < W_L < P$

In the second stage for solving this situation, the usual swap operation cannot resolve it because the part of the neighborhood cannot be searched according to some constraints. The following steps are used to solve this problem.

Step1 Sort the number of set C_L in non-descending order with coil weight.

Step2 Sort the tank j in non-descending order with value of $\sum_{k \in K_j} x_{ij}^k W_i$, where $j \in J \setminus L$.

Step3 Apply pair-swap to enlarge the scale of neighborhood search. **Figure 6** is an example of pair-swap neighborhood for five coils. As shown in the pair swap 1 of Fig. 6, while coil 1 is swapped with coil 2, coil 3 is swapped with coil 4.

Analyzing the character of pair-swap neighborhood, we can get the scale of neighborhood $N(x)$ and the complexity of pair-swap neighborhood is $O(n^4)$.

$$N(x) = (n-1)(n-2)C_2 + (n-2)(n-3)C_2 + (n-3)(n-4)C_2 + \dots + 3 \quad (21)$$

The number of items in formula (21) is $n-3$, where $n \geq 4$ and the last item is equal to 3. Although the scale of pair-swap neighborhood is very large, the number of loaded coils in a tank is not more than nine, and then the search time of

pair-swap neighborhood is acceptable.

4.1.2. Framework of Heuristic Algorithm

For this two-stage heuristic algorithm, the main work is to increase the number of the loaded coils in the second stage in order to solve the difficulty in TS algorithm. The pair-swap operator is very effective to enlarge the search space and obtain a better initial solution at the respect of the number of loaded coils, and the search performance of TS algorithm can also be improved based on this initial solution. **Figure 7** is the flow chart of two-stage heuristic algorithm.

4.2. Neighborhood Structure of TS

Neighborhood definition is a critical issue in the design of many algorithms, and it is important for the tabu search

Fig. 6. The pair-swap neighborhood for five coils.

Fig. 7. Framework of two-stage heuristic algorithm.

in particular. Large neighborhoods generally lead to better local optimal solutions and search accuracy. On the other hand, they also increase the time required to explore the neighborhood at each iteration. Consequently, a larger neighborhood does not necessarily produce a more effective heuristic unless one can explore it in a very efficient manner.

In our algorithm we use two kinds of moves to construct correlative neighborhoods and introduce them as follows.

Swap move: this kind of move includes two instances. (1) One is the commutation of the coils in set C_Y , (2) the other is the commutation between the coil in set C_Y and the coil in set C_L . Take **Fig. 8** as an example, Y_i denotes the i th unloaded coil, L_k^j expresses that the coil is loaded into k th position of j th tank.

K-chains move: this move is a procedure that includes chains of k commutations. The commutation used in move is swap operator, that is to say, if $k=1$, it becomes simple swap move. The neighborhood size of this move increases exponentially along with the size of k , and the computational time needed to explore this large neighborhood also increases rapidly following it. In order to control the computational time effectively, we build a cycle of length $k=2$, just as shown in **Fig. 9**. In this problem, one *K-chains move* is related to commutation of three coils belonging to two or three tanks, and, moreover, one of them may be an unloaded coil in previous move. (a) of **Fig. 9** is the commutation of the loaded coils, and (b) is the commutation with one coil in the set of unloaded coils.

4.3. Tabu List and Stopping Criteria

The size of the tabu list which may be fixed or variable is a very important parameter of tabu search algorithm. It

may be fixed to some values, which means that the tabu list contains a fixed number of prohibited moves. Another kind of tabu list is variable that stochastically adapts to the change of the iterated number in a known bound. This dynamic change is a peculiar disturbance for avoiding the algorithm gets in local minimums. The key is how to confirm the bound of list. In this paper the variable list is adopted to try to produce better results.

In this problem, the tabu list records the assignments of coils and tanks. This prevents the reversal of the assignment status as long as they remain in the list.

The selection of tabu object is based on two distinct neighborhood structures. For swap move, the tabu objects are the numbers, corresponding tanks and positions of two coils. The tabu objects of *k-chains move* are all the corresponding information of three coils.

There are two common stopping criteria used to terminate the search procedure: 1) the maximum number of iterations (t_{max}) has been reached, 2) the maximum number of continuous iterations without improvement (t'_{max}) on the value of the objective function has been reached.

4.4. Diversification Strategy

Besides the standard tabu search feature, additional improving strategies are used to make the tabu search more effective.

In the proposed tabu search algorithm, if the number of continuous iterations without improvement is equal to the given number t_D , the special disturbance is used to mend current solution in order to jump out of local optima. For the convenience of describing the problem, we introduce an imaginary tank as the set of unloaded coils. This kind of disturbance is a k -cycle commutation between some coils in different tanks and the imaginary tank, and k is the number of tank related to change. It is important that the commutations happen between different tanks ensuring the effect of the disturbance. **Figure 10** expresses the disturbance of $k=3$ used in search. Some special areas cannot be explored because of the restriction with some constraints and this disturbance can enlarge the search space effectively.

5. Computational Results

In order to test the performance of the TS algorithm for the train stowage planning problem of coil, we collect the practical data including the tank information and the consolidation planning from transportation department of a practical steel company. For comparison, we solve the model by CPLEX to get the optimal solution and the upper bound

Fig. 8. Two instances of swap move.

Fig. 9. Example of 2-chains move application.

Fig. 10. Example of 3-cycle change application.

(UB) for small problems and for large problems, and we also obtain an upper bound (PUB) of TSPP by relaxing the constraints of stowage requirement to formulate the problem as a knapsack problem. The performance of the improved TS algorithm is proved by comparing with the optimal solution and the upper bound for small-scale problems and comparing with the proposed upper bound for large-scale problems.

The experiments are programmed in visual C++ 6.0 and run on a PC with P4-3.40 GHz CPU and 1 G memory.

5.1. The Upper Bound

In small scale problem, the proposed MILP model can be solved by CPLEX. The improved TS algorithm can be compared by the solutions obtained by CPLEX. However, as the problem scale increases, the computing time for solving the model is increasing rapidly so that up to big-scale problem instances the model cannot be solved within a reasonable time. So we propose an upper bound (PUB) of TSPP to be compared with our TS algorithm as follows.

By relaxing the constraints of stowage requirements, TSPP can be formulated as a knapsack problem. N denotes the total numbers of section and column of the activated coils in the stack yard. The train is supposed to have only one tank, and the number of loading position on the train is n .

$$\text{Maximize } \sum_{i \in C} \sum_{j \in J} \sum_{k \in K} z_3 x_{ij}^k W_i - z_4 N \quad \text{..... (22)}$$

Subject to

$$x_{ij}^k \in \{0,1\} \quad \forall i \in C, \forall j \in J, \forall k \in K \quad \text{..... (23)}$$

$$P \leq \sum_{i \in C} \sum_{k \in K} x_{ij}^k W_i \leq T_j \quad \forall j \in J \quad \text{..... (24)}$$

$$\sum_{i \in C} x_{ij}^k \leq 1 \quad \forall j \in J, \forall k \in K \quad \text{..... (25)}$$

$$\sum_{j \in J} \sum_{k \in K} x_{ij}^k \leq 1 \quad \forall i \in C \quad \text{..... (26)}$$

The activated coils to be loaded onto the train are sequenced according to the non-increasing order of the weight of coils, and we can obtain the set of activated coils, $C' = \{1, 2, \dots, n\}$, where n is the number of coils. It is obvious that $\sum_{j \in J} \sum_{i_1 \in C} \sum_{i_2 \in C} y_{i_1 i_2 j} S_{i_1 i_2} \geq N$, and $-z_4 \sum_{j \in J} \sum_{i_1 \in C} \sum_{i_2 \in C} y_{i_1 i_2 j} S_{i_1 i_2} \leq -z_4 N$.

So the proposed upper bound is the upper bound of TSPP. The activated coils are loaded into the train until weight of the loaded coils subjects to the constraint $P \leq \sum_{i \in C} \sum_{k \in K} x_{ij}^k W_i \leq T_j$. If the activated coils are all loaded onto the train, the upper bound PUB equals to $\sum_{i \in C'} z_3 W_i - z_4 N$. If not all the activated coils are loaded onto the train, the upper bound PUB equals to $\sum_{i \in C'} \sum_{j \in J} \sum_{k \in K} z_3 x_{ij}^k W_i + W_n - z_4 N$. The upper bound can be computed in $O(n \log n)$ time.

5.2. Parameter Setup

The values of the parameters of MILP model are set based on the know-how of the planners and the regulations of train transportation. But the performance of the proposed

algorithm does not depend on the values of the parameters. The maximal value of W_i is 27.5 ton, $T_j=61$, $P=51$, $B_1=1$, $B_2=0.5$, $B_3=13$, the weight coefficients of the object function are $z_1=0.4$, $z_2=0.6$, $z_3=2$, $z_4=1$.

The parameters of algorithm affect the result of computation intensively because it is as delicate as in GA algorithm or SA algorithm. A mass of experiments are done to obtain the appropriate parameters. In this research, the tabu list length is $T_{list} = rand(5, 9)$, the maximal iteration number is $t_{max}=200$, the maximal number of iterations without improvement is $t'_{max}=60$, the number of continuous iterations without improvement is $t_D=20$. It should be noted that, when the value of t_D reaches 20, it is reset to 0, and the value of t'_{max} remains unchanged.

5.3. Experiment Data

All of the experiment data are collected from a transportation department of an iron and steel enterprise. Four instance scales including 5 to 10 coils, 10 to 20 coils, 20 to 50 coils and 50 to 100 coils are tested according the number of activated coils, and each consolidation planning gives the corresponding tank number. In order to indicate the effective of pair-swap neighbourhood in two-stage heuristic, we also take the test with ten different instances.

5.4. Numerical Experiments

We first test the effectiveness of using pair-swap neighbourhood in two-stage heuristic. In Table 2, ten instances are performed to indicate the validity of pair-swap neighborhood. n denotes the number of the activated coils, m denotes the number of loaded tanks, LCN denotes the number of loaded coils in tanks. The experiment results indicate that six instances of them obtain better solution in aspect of the number of loaded coils by pair-swap neighborhood, moreover, the increment of loaded coils number is increasing in evidence along with the change of instance size.

The comparison of solutions obtained by the propose TS with and without the k -cycle change is shown in Table 3. Ten instances are performed to indicate the validity of k -cycle change. LW denotes the value of total weight of the loaded coils in tanks. The experiment results indicate that the loading weight of the train is improved with the k -cycle change.

The comparison of solutions obtained by CPLEX and our heuristics is shown in Table 4. The initial solutions obtained

Table 2. The validity of pair-swap neighborhood.

n	m	LCN with Pair-swap	LCN without Pair-swap
100	4	22	22
100	5	22	21
100	7	30	30
100	8	39	39
100	9	37	36
100	10	40	40
100	11	48	46
200	16	77	75
200	19	93	92
200	22	98	94

Table 3. The validity of k -cycle change.

n	m	LCN without k -cycle change	LCN with k -cycle change	LW without k -cycle change	LW with k -cycle change
100	4	22	22	222.04	222.04
100	5	22	24	278.56	296.75
100	7	30	30	397.11	397.11
100	8	39	38	444.63	451.08
100	9	37	37	505.08	512.32
100	10	40	42	542.90	563.18
100	11	48	48	610.61	611.34
200	16	77	77	897.85	897.85
200	19	93	94	1 056.29	1 069.28
200	22	98	102	1 207.77	1 221.57

Table 4. The comparison between *CPLEX* and our heuristics.

n	m	<i>CPLEX</i>			<i>IS</i>		<i>ISP</i>	
		<i>Obj</i>	<i>Obj</i>	<i>Gap (%)</i>	<i>Obj</i>	<i>Gap (%)</i>	<i>Obj</i>	<i>Gap (%)</i>
5*	1	95.23	94.64	0.62	95.23	0		
10*	2	203.18	195.44	3.81	203.18	0		
17	3	334.23	317.92	4.88	323.41	3.24		
24	6	579.66	525.98	9.26	528.31	8.86		
33	7	821.36	799.12	6.47	799.12	4.30		
39	8	927.36	901.75	2.76	901.75	2.76		
41	9	899.14	843.76	6.16	867.59	3.51		
42	10	1 099	1 053	4.19	1 053	4.19		
48	11	1 278	1 183	7.43	1 229	3.83		
79	16	1 897	1 583	16.55	1 758	7.33		
95	19	2 736	2 417	11.66	2 591	5.3		
100	25	1 981	1 634	17.52	1 822	8.03		

*cases can obtain the optimal solutions by *CPLEX*

without pair-swap neighborhood are shown as *IS*. *ISP* indicates the initial solutions of heuristic with pair-swap neighborhood. *Obj* is the objective value obtained by *CPLEX* or our heuristics. *Gap* denotes the gap between the solutions obtained by heuristics and those obtained by *CPLEX*. The value of *Gap* is calculated by the formula as follows:

$$Gap (\%) = \frac{CPLEX - Obj}{CPLEX} \times 100\%$$

CPLEX denotes the optimal solution or the upper bound obtained by *CPLEX*. From Table 4, we can conclude that initial solutions generated in *ISP* are better than *IS*. The gap increases along with the change of instance size, however, the average gap of *ISP* is smaller than that of *IS*. This means that the two-stage heuristic with pair-swap neighborhood can obtain a better initial solution. The average gaps of *IS* and *ISP* are 7.61% and 4.28%.

The heuristic is proposed according to the know-how of the planners in practice, and the practical data of the manual operation is difficult to calculate. So we use the solutions obtained by heuristic *IS* to indicate the solutions obtained by manual operation roughly. The average loading rate and average loading time obtained by manual operation are about 96.22% and 1 200 seconds, and those obtained by tabu

Table 5. The comparison of the solutions obtained by manual operation and by the tabu search.

n	m	<i>IS</i> (by manual operation)		Tabu search	
		loading rate (%)	Time (s)	loading rate (%)	Time (s)
5	1	95.73	1 200	98.28	0.53
7	2	96.11	1 200	99.72	1.21
10	2	96.17	1 200	99.49	3.99
14	3	96.23	1 200	99.82	4.50
20	4	96.95	1 200	99.67	4.23
30	8	96.26	1 200	99.89	5.27
40	10	96.03	1 200	99.59	8.64
50	12	96.31	1 200	99.15	18.71
Average		96.22	1 200	99.45	5.89

Table 6. The experiment results of coil number from 5 to 10.

n	m	<i>CPLEX</i>				Tabu search		
		<i>Obj</i>	<i>UB</i>	<i>Gap (%)</i>	Time (s)	<i>Obj</i>	<i>Gap (%)</i>	Time (s)
5*	1	95.23	95.23	0	5.4	95.23	0	0.53
6*	2	98.16	98.16	0	6.61	98.16	0	1.01
7*	2	98.16	98.16	0	6.61	98.16	0	1.21
8*	2	99.14	99.14	0	9.02	99.14	0	2.02
9*	2	102.24	102.24	0	12.03	102.24	0	3.52
10*	2	203.18	203.18	0	15.05	203.18	0	3.99
Average		116.02	116.02	0	9.12	116.02	0	2.05

*cases can obtain the optimal solutions by *CPLEX*

search are about 99.45% and 5.89 seconds. *Time* denotes the time of obtaining solution. *loading rate* denotes the loading rate of the train. Comparing to the solutions obtained by planners, the average computing time of the tabu search is decreased significantly and the loading rate of the train is increased by 3.23%. The value of *loading rate* is calculated by the formula as follows:

$$loading\ rate (\%) = \frac{loading\ weight}{loading\ capacity\ of\ train} \times 100\%$$

We use *CPLEX* to obtain the optimal solution or best found solution to compare with our results obtained by the tabu search algorithm. *Time* is the time of obtaining optimal solution for *CPLEX* and the tabu search heuristic. For small-scale problems, we evaluate the effectiveness of our tabu search algorithm by comparing with the solution obtained by the standard software *CPLEX* in **Tables 6** and **7**. The computation time of the standard software is limited to 60 min. If the cases of *CPLEX* have ended in the calculation time limitation, it means that the optimal solutions are obtained by it. And the optimal solutions are compared with the solutions obtained by the proposed tabu search. Otherwise, the upper bounds obtained by *CPLEX* are compared with the solutions obtained by the proposed tabu search.

For large-scale problems, the problem cannot be solved by *CPLEX*, and the proposed upper bounds *PUB* are compared with the solutions obtained by the improved TS algorithm in **Tables 8** to **9**.

For the small-scale instances denoted in Table 6, *CPLEX*

Table 7. The experiment results of coil number from 10 to 20.

<i>n</i>	<i>m</i>	<i>CPLEX</i>				Tabu search		
		<i>Obj</i>	<i>UB</i>	<i>Gap (%)</i>	<i>Time (s)</i>	<i>Obj</i>	<i>Gap (%)</i>	<i>Time (s)</i>
11*	3	213.82	213.82	0	28.81	213.82	0	4.23
13	3	214.58	287.36	25.30	3 600	268.32	6.62	4.21
14	3	299.30	322.74	7.26	3 600	303.23	6.05	4.50
17	3	311.30	334.23	6.86	3 600	323.41	3.24	4.51
19	3	323.26	341.17	5.25	3 600	334.31	2.01	4.67
20	4	244.10	476.62	48.79	3 600	448.26	5.95	4.23
Average		267.73	329.32	15.58	3 005	315.23	3.98	4.39

*cases can obtain the optimal solutions by *CPLEX*

Table 8. The experiment results of coil number from 20 to 50.

<i>n</i>	<i>m</i>	<i>PUB</i>	Tabu search		
			<i>Obj</i>	<i>Gap (%)</i>	<i>Time (s)</i>
23	5	551.65	522.44	5.30	4.50
24	6	589.66	572.80	2.86	4.51
25	6	604.41	565.31	4.67	3.99
26	6	592.76	542.47	8.48	4.62
28	7	724.71	693.33	4.33	4.65
30	8	878.49	850.73	3.16	5.27
32	8	892.47	864.55	3.13	5.26
33	8	911.70	890.63	2.31	5.22
36	9	917.44	888.71	3.15	10.20
38	9	1 018.36	948.70	6.84	13.24
40	10	1 021.94	987.11	3.41	8.64
43	11	1 112.56	1 021.52	8.18	14.4
45	11	1 243.74	1 201.15	3.42	16.23
48	12	1 286.54	1 207.12	6.17	18.11
50	12	1 310.30	1 240.87	5.30	18.71
Average		910.45	866.50	4.71	9.17

Table 9. The experiment results of coil number from 50 to 100.

<i>n</i>	<i>m</i>	<i>PUB</i>	Tabu search		
			<i>Obj</i>	<i>Gap (%)</i>	<i>Time (s)</i>
60	15	1 537.01	1 446.63	5.88	20.61
70	18	1 775.43	1 618.30	8.85	21.3
80	18	1 839.25	1 718.31	6.58	21.9
90	20	2 113.77	2 006.58	5.07	22.78
100	20	2 365.15	2 175.23	8.03	24.25
Average		1 926.12	1 793.01	6.88	22.17

and our algorithm can obtain optimal solutions in a short time, but our algorithm cost less time. In Table 7, *CPLEX* cannot obtain the optimal solutions in a reasonable time, and the solutions obtained by our algorithm are compared by the upper bound *UB* obtained by *CPLEX*. For the large-scale

instances denoted in Tables 8 and 9, our algorithm gets better solutions than those obtained by the upper bounds *PUB* by relaxing the constraints of stowage requirements. The average *Gap* between the solutions of our algorithm and the upper bound *PUB* is less than 10%. On the other hand, the time that our algorithm used is less than 60 seconds. Therefore, our algorithm can solve TSPP effectively in practical loading process.

6. Conclusion

A multi-tank train stowage planning problem is studied in this paper, which is formulated to be a novel MILP model. A two-stage heuristic algorithm according to the know-how of the planners in practice is proposed to obtain an initial solution for tabu search algorithm. Considering the characteristics of the problem, the tabu search algorithm with different neighbourhoods and diversification strategy are proposed. The solutions obtained by our algorithm are compared with the optimal solutions and the upper bound obtained by *CPLEX* in small-scale problems, and compared with the proposed upper bound of the problem in large-scale problems. The computational results show that our algorithm can obtain near-optimum solutions in small scale problems and is more effective than *CPLEX* in large scale problems in both the quality of solutions and the computing time. In practice, the train stowage plan is made manually based on the planner's know-how. The loading rate is lower by 3.23% than our algorithm and the loading time is longer, and the second objective of our model is not considered by the planners, which will cause several unnecessary operations of cranes. The research in this paper can be used to direct the loading operation in practice effectively.

Acknowledgement

This research is partly supported by State Key Program of National Natural Science Foundation of China (Grant No. 71032004) and the Foundation for Innovative Research Groups of the National Natural Science Foundation of China (Grant No. 71321001) and State Key Laboratory of Synthetical Automation for Process Industries Fundamental Research Funds (Grant No. 2013ZCX04-01).

REFERENCES

- 1) T. Umeda, A. Kitamura, M. Konishi, S. Kanamura and S. Takami: *ISIJ Int.*, **41** (2001), No. 5, 446.
- 2) M. Avriel, M. Penn, N. Shpirer and S. Witteboon: *Ann. Oper. Res.*, **76** (1998), 55.
- 3) D. Ambrosino, A. Sciomachen and E. Tanfani: *Transport. Res., A*, **38** (2004), 81.
- 4) A. Imai, K. Sasaki, E. Nishimura and S. Papadimitriou: *Eur. J. Oper. Res.*, **171** (2006), 373.
- 5) J. F. Álvarez: *J. Marit. Res.*, **3** (2006), No. 1, 3.
- 6) L. X. Tang, R. Zhao and J. Y. Liu: *Nav. Res. Log.*, **59** (2012), No. 7, 502.
- 7) F. Glover: *Comput. Oper. Res.*, **13** (1986), 533.
- 8) P. Hansen: Cong. on Numerical Methods in Combinatorial Optimization, (1986), 1.
- 9) E. Nowicki and C. Smutnicki: *Eur. J. Oper. Res.*, **106** (1998), 226.
- 10) G. Barbarosoglu and D. Ozgur: *Comput. Oper. Res.*, **26** (1999), 255.
- 11) J. P. Watson, A. E. Howeb and L. D. Whitley: *Comput. Oper. Res.*, **33** (2006), 2623.
- 12) B. A. Norman: *Comput. Oper. Res.*, **36** (1999), 163.
- 13) J. Grabowski and M. Wodecki: *Comput. Oper. Res.*, **31** (2004), 1891.