An Integrated Optimization Method for Heavy Haul Trains with Virtual Coupling Based on Genetic Algorithm

Lezhou Wu * **Hao Ye** *,1 **Zhihua Xiong** * **Wei Dong** **

* *Department of Automation, Tsinghua University, Beijing, PR China (e-mail: haoye@tsinghua.edu.cn)*
** *Beijing National Research Center for Information Science and Technology, Tsinghua University, Beijing, PR China*

Abstract: Heavy haul transportation characterized by long trains enjoys the benefits of cost-efficiency but suffers from high cost in combination process. Virtual coupling (VC) is a state-of-art train control technology that can help resolving the dilemma. In this paper, we proposed an integer programming model that schedules station operation plans, timetables and train combination schemes of heavy haul trains (HHTs) with VC at the same time. A genetic algorithm with a deep first search decode method is proposed to efficiently solve this problem on a large scale. The simulation results show that our method can effectively improve the efficiency of HHTs in real-world scenarios.

Keywords: Heavy haul transportation, virtual coupling, optimal scheduling, integer programming, genetic algorithm.

1. INTRODUCTION

Heavy haul transportation is a widely adopted freight transportation method, because of its outstanding capacity for regional bulk cargo transport and merits of cost-efficiency and punctuality Davydov et al. (2017); Zhou et al. (2021). However, the usage of heavy haul trains (HHTs) with extreme lengths bring difficulty to train combination processes, as the combination plans are complex and track resources of technical stations are limited Zhou et al. (2021). So, in practice, stations operation plans (i.e., loading/unloading job plans), timetables and combination schemes are typically created separately to reduce complexity Zhou et al. (2021), which cannot guarantee the quality of the plans.

Virtual coupling (VC) is an advanced technology for train control system based on vehicle-to-vehicle communication Wu et al. (2023), which makes it possible to establish a long train by virtually combining many short unit trains, with each unit train still running independently. VC technology can significantly reduce the cost of combination processes and the complexity of train combination plans, making it possible to schedule the station operation plans, timetables and train combination plans in an integrated way, and obtain a scheduling solution closer to the global optimum. So in this paper, the integrated scheduling problem based VC will be discussed.

Most existing works related to VC focus on the control methods Wu et al. (2023) and the operation logic Di Meo et al. (2020) of VC. Only a few recent papers addressing the scheduling problems for trains with VC technology. In Chen et al. (2022), the authors consider both the dynamics of trains and passenger flow and propose a joint state-space model integrated with VC to schedule trains in a metro line. Chai et al. (2023) takes a further step and considers a more complex metro network. Both Chen et al. (2022) and Chai et al. (2023) assume that trains can only be combined at the beginning of the trip. Wang and Su (2022) schedules metro trains with VC on a Y-shaped circle line, to handle the unbalanced passenger demand. Chen et al. (2024) also investigates the train scheduling problem on Y-shaped lines using VC, and they not only optimize the passenger flow and total train delays, but also analyses the switching sequence of turnouts considering safety. In Zhou et al. (2023), the timetable rescheduling problem on high-speed railway lines with VC is solved. All the works related to scheduling mentioned above are related to passenger trains scheduling, where station operations plans are neglected and the numbers of trains considered are far smaller than that in heavy haul transportation (typically hundreds of trains). A recent work Ma et al. (2024) has considered the timetable rescheduling problem of HHTs based on VC, they also fail to consider the station operation plans and the scale of their problem remains far smaller than that of the HHT scheduling problem. Our previous work Wu et al. (2024) provided a detailed integer programming model for scheduling HHTs with VC, which can be applied to the integrated problem in this paper, but it is not capable of handling large-scale scheduling problem. Therefore, the existing methods are not applicable if we want to schedule the large-scale station operation plans, timetables and train combination plans in an integrated way.

In this paper, we propose an integrated model that considers station operation plans, timetable and VC combination schemes at the same time. With the objective function that aims at improving the transportation efficiency, the model

[1] Corresponding author.

is organized as an integer programming (IP) problem with discrete time variables. Considering the large number of trains in real scenarios and the complexity caused by the nonlinear parts of the IP problem, we propose a genetic algorithm (GA) algorithm with an deep first search (DFS) decode method to solve it effectively.

The rest of the paper is organized as follows: section 2 describes the problem and formulates it into an IP problem; section 3 provides the details of our proposed GA methods; section 4 conducts simulation experiments under the real-scenario settings to verity the effectiveness and efficiency of our proposed method; in section 5, we conclude the paper and provide some directions for further research.

2. PROBLEM FORMULATION

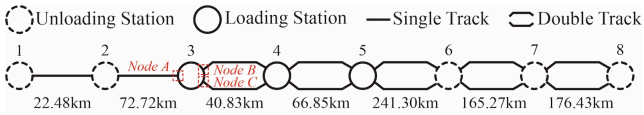

Fig. 1. Heavy haul railway line

The problem considered in this paper can be described as: given the railway line information and the origin and destination stations for the trains such as the example given in Fig. 1, we need to make the following decisions: (i) the selection of the segments (Y_f^l/Y_f^u) for loading and unloading operations and the specific periods ($C_f^{L/U}/S_f^{L/U}$) for these operations; (ii) when a train should arrive at which station ($t_{f,\aleph}$), i.e., timetables; (iii) at which stations and among which trains the combination should occur ($\xi_{f,f',\aleph}$), i.e., combination schemes. The scope of the decisions should cover the entire process from when heavy trains (i.e., trains with cargo) depart from the loading stations (to the unloading stations) until corresponding empty trains (i.e., trains without cargo) return from the unloading stations (to the loading stations). The railway line includes both single-track lines which allow trains to run in both directions, and double-track lines where each track is dedicated to one direction. Moreover, all the decisions should follow constraints of train operations. To simplify the problem, following assumptions are followed: (i) the shunting time of trains in the stations is included in the loading/unloading operation time; (ii) the stations have sufficient tracks available for trains to wait for loading/unloading operations; (iii) the train length and acceleration/deceleration processes are neglected; (iv) the position where a station connects to an interstation track is referred to as a node (as shown in Fig. 1), and trains passing through different nodes do not interfere with each other, so our timetable scheduling will base on nodes. It is worth noting that Fig. 1 just provides an example of a real railway line, but the model for problem formulation and the proposed method discussed in the following sections do not rely on the specific structure of Fig. 1.

We note that station operations and train sequencing planning share similarities with flexible job shop scheduling problems (FJSSP) described in Chaudhry and Khan (2016). Therefore, we adopts some of the FJSSP formulations, and integrate them with existing constraints related to scheduling trains with VC Wang and Su (2022); Wu et al. (2024); Zhou et al. (2023) and constraints unique to our problem to form the IP model of this paper.

2.1 Loading and Unloading Operations Constraints

A heavy train can only choose one and only loading and unloading segment respectively, i.e.,

$$\sum_{l\in L_f} Y_f^l = 1, \forall f \in F_h \tag{1}$$

$$\sum_{u\in U_f} Y_f^u = 1, \forall f \in F_h \tag{2}$$

where F_h is the set of heavy trains, L_f/U_f is the set of loading/unloading segments train f can choose and $Y_f^l/Y_f^u = 1(0)$ represents train f chooses (does not chose) segment l as its loading/unloading segment.

The start time of the loading job must be non-negative, and the unloading job must start after the heavy train arrives at its unloading station, which can be respectively represented as

$$S_f^L \geq 0, S_f^U \geq t_{f,D_f}, \forall f \in F_h \tag{3}$$

where S_f^L/S_f^U is the time when the train f starts the loading/unloading job, t_{f,D_f} is the time when train f arrives at its unloading node D_f. The loading/unloading duration is related to the choice of loading/unloading segment, i.e.,

$$C_f^L - S_f^L = \sum_{l\in L_f} Y_f^l \cdot \tilde{t}_l, \forall f \in F_h \tag{4}$$

$$C_f^U - S_f^U = \sum_{u\in U_f} Y_f^u \cdot \tilde{t}_u, \forall f \in F_h \tag{5}$$

where C_f^L/C_f^U is the time when train f completes the loading/unloading job, and $\tilde{t}_l/\tilde{t}_u$ is the time loading/unloading segment l/u needs to finish the job.

When two trains choose the same loading/unloading segment, there is one and only one precedence order between them, $\forall f, f' \in F_h, f \neq f'$

$$P_{f,f',w} + P_{f',f,w} \geq Y_f^w + Y_{f'}^w - 1 \tag{6}$$

$$P_{f,f',w} + P_{f',f,w} \leq 1 \tag{7}$$

$$\forall w \in \{l, u \mid l \in L_f \cap L_{f'}, u \in U_f \cap U_{f'}\}$$

where $P_{f,f',w} = 1(0)$ represents train f starts its loading/unloading job earlier (later) than train f' on segment w. Moreover, two trains cannot occupy the same segment simultaneously for work, which can be represented with $P_{f,f',w}$ as, $\forall f, f' \in F_h, f \neq f'$

$$C_{f'}^L - C_f^L + M \cdot (3 - P_{f,f',l} - Y_f^l - Y_{f'}^l) \geq \tilde{t}_l \tag{8}$$

$$C_{f'}^U - C_f^U + M \cdot (3 - P_{f,f',u} - Y_f^u - Y_{f'}^u) \geq \tilde{t}_u \tag{9}$$

$$\forall l \in L_f \cap L_{f'}, \forall u \in U_f \cap U_{f'}$$

where M is a large enough number.

Remark 1. The forms of (1)-(9) are based on commonly used formulation of FJSSP Chaudhry and Khan (2016).

2.2 Timetable Constraints

Heavy trains must depart from their loading stations only after the loading jobs are completed, and empty trains must depart from corresponding unloading stations only after the unloading jobs are completed. These constraints can be represented as

$$t_{f,O_f} \geq C_f^L, \forall f \in F_h \tag{10}$$

$$t_{f,O_f} \geq C_{f'}^U - M \cdot (1 - r_{f,f'}), f \in F_e, f' \in F_h \tag{11}$$

where O_f is the original node of train f, F_e is the set of empty trains, $r_{f,f'}$ means whether train f and f' share the same rolling stock, i.e., whether heavy train f' becomes empty train f after its unloading job. A minimum time is required for a train to travel between two nodes due to speed limitations, i.e., Wang and Su (2022)

$$t_{f,s_{f,k+1}} - t_{f,s_{f,k}} \geq FT(s_{f,k}, s_{f,k+1}) \tag{12}$$
$$\forall f \in F, \forall s_{f,k}, s_{f,k+1} \in V_f$$

where $s_{f,k}$ is the kth node that train f run through, $FT(i,j)$ is the minimum time for trains to run from node i to j, $F = F_e \cup F_h$ and V_f is the set of nodes train f runs through.

To discuss the relationships between different trains in the timetable, we first need to determine the sequence in which trains run through each node, which can be represented by the following set of constraints, $\forall f, f' \in F, f \neq f'$

$$\sum_{m=1}^{n_{s_{f,k}}} X_{f,s_{f,k},m} = 1, \forall s_{f,k} \in V_f \tag{13}$$

$$\sum_{f,\aleph \in V_f} X_{f,\aleph,m} \leq 1, \forall \aleph \in V, m = 1, \cdots, n_\aleph \tag{14}$$

$$X_{f,s_{f,k+1},m} = X_{f,s_{f,k},m}, \forall (s_{f,k}, s_{f,k+1}) \in E_f \tag{15}$$

$$p_{f,f',\aleph} = \sum_{m=1}^{n_\aleph - 1} \left(X_{f,\aleph,m} \cdot \sum_{l=m+1}^{n_\aleph} X_{f',\aleph,l} \right) \tag{16}$$
$$\forall \aleph \in V_f \cap V_{f'}$$

where $X_{f,\aleph,m} = 1(0)$ means train f is (not) the mth train to pass through node $\aleph$, $n_\aleph$ is the number of trains that run through node $\aleph$, E_f is the set of interstation tracks train f runs through, $V = \cup_{f \in F} V_f$ and $p_{f,f',\aleph} = 1(0)$ represents train f runs through node $\aleph$ earlier (later) than f'. Equation (13) ensures that each train is uniquely assigned to one sequential position of a node. Equation (14) means each sequential position of a node can be assigned to at most one train. Equation (15) means that trains cannot overtake each other on the interstation tracks. Equation (16) establishes the relationship between $X_{f,\aleph,m}$, $X_{f',\aleph,l}$ and $p_{f,f',\aleph}$.

Remark 2. The formulations of (13) and (14) is inspired by the sequencing representations in FJSSP Roshanaei et al. (2013).

Considering the safety requirement, there is a minimum headway between two trains, which is related to the VC mode and can be written as Wu et al. (2024)

$$t_{f',\aleph} - t_{f,\aleph} + (1 - p_{f,f',\aleph}) \cdot M \geq \xi_{f,f',\aleph} \cdot t_{VC} + (1 - \xi_{f,f',\aleph}) \cdot t_{NVC}, \forall f, f' \in F, f \neq f', \forall \aleph \in V_{f'} \cap V_f \tag{17}$$

where $\xi_{f,f',\aleph} = 1(0)$ represents train f and f' are (not) virtually combined on node $\aleph$, and headways are t_{VC} and t_{NVC} in the two modes ($t_{VC} < t_{NVC}$). And $p_{f,f',\aleph} = 1(0)$ represents train f arrives at node $\aleph$ earlier (later) than train f'.

2.3 Virtual Coupling Constraints

Due to the requirement of VC technology, several related constraints should be satisfied. First, VC relation can only be established between two trains running in the same direction

$$\xi_{f,f',\aleph} \leq 1 - \mid x_{f,\aleph} - x_{f',\aleph} \mid \tag{18}$$
$$\forall f, f' \in F, f \neq f', \aleph \in V_f \cap V_{f'}$$

where $x_{f,\aleph} = 1(0)$ represents train f heads in the up (down) direction when passing through node $\aleph$. Second, the headway between two virtually combined trains cannot be too big considering the minimum communication distance constraint Zhou et al. (2023)

$$(\xi_{f,f',\aleph} - 1) \cdot M + t_{f',\aleph} - t_{f,\aleph} \leq T_{VC}^{MAX} \tag{19}$$
$$\forall f, f' \in F, f \neq f', \forall \aleph \in V_{f'} \cap V_f$$

where T_{VC}^{MAX} is the maximum headway between two virtually combined trains. Third, the number of trains in a VC group (the number of consecutively virtually combined trains) cannot exceed N_c, i.e.,

$$\sum_{i=0}^{N_c - 1} c_{m+i,\aleph} \leq N_c - 1 \tag{20}$$
$$\forall \aleph \in V, m = 1, 2, \cdots, n_\aleph - N_c$$

where $c_{m,\aleph} = 1(0)$ means the mth train that runs through node $\aleph$ is (is not) virtually combined with the m+1th train. Finally, the relation between variable $c_{m,\aleph}$ and $\xi_{f,f',\aleph}$ is established through the following constraint

$$\xi_{f,f',\aleph} = \sum_{m=1}^{n_\aleph - 1} (X_{f,\aleph,m} \cdot X_{f',\aleph,m+1} \cdot c_{m,\aleph}) \tag{21}$$
$$\forall f, f' \in F, f \neq f', \forall \aleph \in V_f \cap V_{f'}$$

Remark 3. In subsection 2.2 and 2.3, constraints (10) (11) (15) (16) (18) (20) (21) are unique to our problem.

2.4 Objective Function

For heavy haul transportation, improving cargo transportation speed and train turnover efficiency is an important objective. Therefore, our objective function is defined as follows

$$\min OBJ = \sum_{f \in F_e} t_{f,D_f} \tag{22}$$

Obviously, this objective function minimizes the sum of the times for all trains to return to their origin stations. When the cargo volume is fixed, minimizing OBJ is equivalent to maximizing transportation efficiency. In the following text, $-OBJ$ is also used as the fitness function in the GA method, i.e., larger fitness value is better.

3. SOLUTION METHOD BASED ON GA

The integer programming model proposed in section 2 has an NP-hard nature, with solution complexity increasing exponentially as the problem size grows. Therefore, the large-scale practical scheduling problem of heavy haul trains with VC (involving hundreds of trains and containing up to 1M decision variables) cannot be solved directly with commercial solvers, e.g., GUROBI solver. Inspired by the flexible job shop scheduling problem Chaudhry and Khan (2016), we notice that the heavy trains traveling to the destination stations for unloading and the empty trains returning to the origin stations can be seen as two operations, while the selection of loading and unloading segments can be seen as the choice of processing machines in a job shop. So in this section, we adopt the GA method widely used in FJSSP, and propose an efficient decode strategy tailored to our problem, i.e., DFS decode method, to solve the problem effectively. The main GA iteration process is the normal GA workflow with elite reservation strategy. Readers can refer to the related works Amjad et al. (2018) for details.

3.1 Encode, Crossover and Mutate

To encode the decision variables, we construct a train sequence chromosome ($\mathcal{C}_{seq}$) and a loading/unloading segment dispatch chromosome ($\mathcal{C}_{dis}$), both with a length of $|F_h \cup F_e|$. For the former, the ID of each train appears twice: the first appearance represents its scheduling order in the heavy train state, and the second time represents its scheduling order in the empty train state. For the latter, the $2i-1$th and $2i$th element of the chromosome represent the loading and unloading segment dispatch for the ith heavy train, respectively. We implement crossover operation of the two parts of chromosomes separately to ensure the feasiblility of child chromosomes. For $\mathcal{C}_{seq}$, improved precedence operation crossover (IPOX) is utilized, while for $\mathcal{C}_{dis}$, we use multipoint preservative crossover (MPX). Interested readers may refer to Amjad et al. (2018) for further details. Similarly, different mutation techniques are employed for two chromosomes. For $\mathcal{C}_{seq}$, we randomly chose a position and a gene in the chromosome, then move the selected gene to the chosen position. For $\mathcal{C}_{dis}$, we random select an index, and replace the segment number in the selected index by another segment number from the same alternative segment set. The above operations are repeated a given times during the mutation process. The crossover and mutation processes occur with the probability p_c and p_m respectively in each GA iteration.

3.2 DFS Decode Method

Inspired by the experience in FJSSP, compared with the passive decode method, where the sequence of trains in the timetable corresponds directly to the order represented in $\mathcal{C}_{seq}$, the active decode method, where the timetable of a given train is set to the earliest feasible time stamp, is more effective. However, when constructing the timetable, there are more complex constraints between trains and greedily choosing the earliest idle time stamps can often lead to an infeasible solution. For example, in Fig. 2, the time interval b and c contain feasible time stamps for the currently

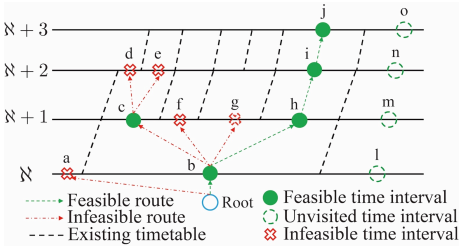

Fig. 2. Schematic of the DFS decode method

Algorithm 1 DFS Decode Method

Input: Train sequence $\mathcal{C}_{seq}$, segment dispatch $\mathcal{C}_{dis}$

Output: Timetable $\mathcal{T}$, loading time of heavy trains $\mathcal{L}$, unloading time of heavy trains $\mathcal{U}$, fitness $\mathcal{F}$

1: $\mathcal{T} \leftarrow \emptyset$, $\mathcal{L} \leftarrow \emptyset$, $\mathcal{U} \leftarrow \emptyset$, $\mathcal{F} \leftarrow 0$ ▷ Initialize the result
2: **for** f_c in $\mathcal{C}_{seq}$ **do**
3: **if** $f_c \in F_h$ **then**
4: Arrange loading time of f_c in $\mathcal{L}$ according to $\mathcal{C}_{dis}$
5: **end if**
6: $\mathcal{T}_{f_c} \leftarrow \text{DFS}(f_c, s_{f_c,1})$ ▷ Get timetable of train f_c
7: Insert $\mathcal{T}_{f_c}$ to current timetable $\mathcal{T}$
8: Get the time $t_{f_c,D_{f_c}}$ when train f arrives at its destination
9: **if** $f_c \in F_h$ **then**
10: Arrange unloading time of f_c in $\mathcal{U}$ according to $\mathcal{C}_{dis}$ and $t_{f_c,D_{f_c}}$
11: **else**
12: $\mathcal{F} \leftarrow \mathcal{F} - t_{f_c,D_{f_c}}$ ▷ Update fitness according to objective (22)
13: **end if**
14: **end for**
15: **function** DFS(f_c,$s_{f_c,k}$)
16: $\mathcal{T}_{f_c} \leftarrow \emptyset$ ▷ Initiate sub-timetable of train f_c
17: Find set S contains all the time interval of node $s_{f_c,k}$
18: **for** s in S **do**
19: Find set T contains all the feasible time stamps in s according to constraints (10-12) (17-20)
20: **if** $T \neq \emptyset$ **then**
21: $t_I \leftarrow \min(T)$ ▷ Greedily choose the earliest time stamp
22: **if** $s_{f_c,k}$ is the destination node **then**
23: $\mathcal{T}_{f_c} \leftarrow t_I$
24: **return** $\mathcal{T}_{f_c}$
25: **end if**
26: $\mathcal{T}^s_{f_c} \leftarrow \text{DFS}(f_c, s_{f_c,k+1})$ ▷ Get sub-timetable recursively
27: **if** $\mathcal{T}^s_{f_c} \neq \emptyset$ **then**
28: $\mathcal{T}_{f_c} \leftarrow [t_I|\mathcal{T}^s_{f_c}]$ ▷ Concat time stamps
29: **return** $\mathcal{T}_{f_c}$
30: **end if**
31: **end if**
32: **end for**
33: **return** $\mathcal{T}_{f_c}$
34: **end function**

Fig. 3. DFS decode method algorithm

considered train and will be chosen if the traditional active decode method used in FJSSP is adopted, but choosing b and c will lead to an infeasible solution as d and e do not contain feasible time stamps and other time intervals cannot be chosen because overtaking is not allowed in interstation segment $(\aleph+1, \aleph+2)$.

Noticing that the choice of the time interval is valid only if selecting this time interval can result in a complete timetable of the currently scheduled train, we propose a DFS decode method to recover the timetable and loading/unloading time scheduling results from any given valid $\mathcal{C}_{seq}$ and $\mathcal{C}_{dis}$, as shown in Fig. 3.

The core of the DFS decode method is the DFS method to find the feasible timetable for the currently considered train f_c in step 6 and is listed out in detail in step 15-34 in Fig. 3. DFS recursively searches each node train f_c runs

through, i.e., $s_{f_c,k}$, find the feasible time interval set (step 17) and time stamp set (step 19) of the node, and choose the earliest time stamp in the feasible time stamp set (step 21) to add to the timetable of the f_c (step 28). The search process stops as soon as the first complete timetable of f_c is gotten. In the example in Fig. 2, the order in which DFS visits the time intervals is a-b-c-d-e-f-g-h-i-j, the final timetable of f_c is in the time intervals b-h-i-j, and the time intervals l-m-n-o will not be visited as the complete timetable has been gotten. A feasible timetable can always be found as the last time interval of each node has an infinite duration. Moreover, in step 4 and step 10, the loading and unloading times of the train f_c are arranged in the earliest feasible time interval of the loading and unloading segments selected by $\mathcal{C}_{dis}$.

4. EXPERIMENTAL RESULTS

In this section, we test our method under a real railway line scenario which contains 3 loading stations and 5 unloading stations is studied (as shown in Fig. 1). The speed limit of trains in the line is 80 km/h, $t_{VC} = 3$ min, $t_{NVC} = 6$ min, $T_{VC}^{MAX} = 5$ min, $N_c = 10$ and the detailed information of loading/unloading segment working times is listed in Table 1, where m*n means there are n segments with the working time of m minutes. In subsection 4.1, we validate the effectiveness of the method on small-scale problems, and in subsections 4.2, we evaluate the efficiency of our method on large-scale problems. We conduct experiments on the personal computer with i9-13900k CPU and 64GB DDR4 RAM, and all the algorithms are implemented with MATLAB 2024a or GUROBI v11.0.3.

Table 1. Station information

Station	Type	Segment working time (min)
1	Unloading	240*3/2880*3/780
2	Unloading	300*4
3	Loading	3000*5
4	Loading	180*10/160/150*3/140/120*3
5	Loading	240*2
6	Unloading	730/654/297
7	Unloading	375
8	Unloading	90*13

4.1 Effectiveness of the Method

In this subsection, we verify the quality of the solutions obtained by our method through some small-scale problems. Specifically, we compare the solution quality as well as the solving speed of our method with those of GUROBI solver. In these experiments, the solving time of GUROBI is limited to 3600 s, and the parameters of our method are population size $N_P = 100$, iteration number $N_I = 200$, elite individual number $N_{top} = 20$, $p_c = 0.8$, $p_m = 0.2$ (in the following text, unless otherwise specified, these parameters remain unchanged). The results are shown in Table 2 (SCE=scenario, N_T=train number, GUR=GUROBI, RE=results, RT=runtimes (s)), where the scenario m×n means m loading stations and n unloading stations are considered and the better results are marked in bold. In each problem, we conduct 10 times of experiments with our method and show the mean value of these experiments.

On one hand, the mean results achieved by our method are no worse than those of GUROBI in most experiments and

Table 2. Results of small-scale problems

SCE	N_T	GUR RE	GUR RT	Our RE	Our RT
1×1	10	13023	3600.09	**12531.00**	**25.54**
1×1	15	19435	3600.29	**19048.10**	**34.26**
2×2	10	**10502**	3600.05	10506.60	**48.36**
2×2	15	-	3600.03	**16894.60**	**70.69**
3×3	10	**16602**	465.42	16602.80	**40.25**
3×3	15	57972	3600.09	**25680.20**	**61.48**
3×5	10	**16291**	99.26	**16291.00**	**38.05**
3×5	15	24037	3600.06	**24026.60**	**61.16**

are only slightly inferior to the results of GUROBI in a few cases. This observation indicates that the formulation of the chromosome and the heuristics used in the decoding process do not result in significant loss of optimality, and the effectiveness of the proposed method is thus verified. On the other hand, the runtimes of our method do not experience significant variation with changes in problem size and is notably faster than that of GUROBI in all cases. Moreover, GUROBI's solving time can increase rapidly with problem size, and in scenario 2×2 with 15 trains, it even fails to obtain a feasible solution.

4.2 Efficiency of the Method

To demonstrate the efficiency of our method, we compare it with first come first serve (FCFS), pure GA (PGA) and tabu search (TS). The encode methods of PGA and TS are the same as our method, and both use passive decode method to recover the final results. The parameters of PGA are consistent with those of our method, while for TS, its iteration number is 200, number of neighbors is 20 and tabu length is 10. The objective values and runtimes (in the bracket) of each method for different problem sizes are shown in Table 3 (the following experiments consider all the stations), where each value is the average of 10 experiments, and the best results are highlighted in bold.

Table 3. Solutions and runtimes of different methods under varying problem scales

Method	30 trains	60 trains	90 trains	120 trains	150 trains
FCFS	91635 (**0.01**)	206817 (**0.02**)	269004 (**0.04**)	399410 (**0.06**)	626930 (**0.08**)
PGA	42183 (29.9)	115340 (54.9)	188127 (94.9)	296911 (131.1)	489826 (160.8)
TS	42279 (38.9)	107535 (81.3)	173234 (149.3)	269630 (214.0)	446023 (276.6)
Ours	**36675** (139.9)	**84717** (269.7)	**137814** (403.3)	**201100** (500.0)	**284534** (601.8)

For one thing, the optimization results of our method are significantly better than all other comparison methods, especially when the car flow is more complex. For another, the computational time of our method is longer than that of PGA and TS methods, because the computational complexity for constructing the timetable of a train with our DFS method is $O(|S_{all}|)$ where $|S_{all}|$ is the number of time intervals in the current timetable, while the corresponding computational complexity of the passive decode method is $O(1)$. However, as shown in Fig. 4 where a/b means N_P/N_I in the GA settings, under all the parameter settings, the computational times of our method increase linearly with the train number (without

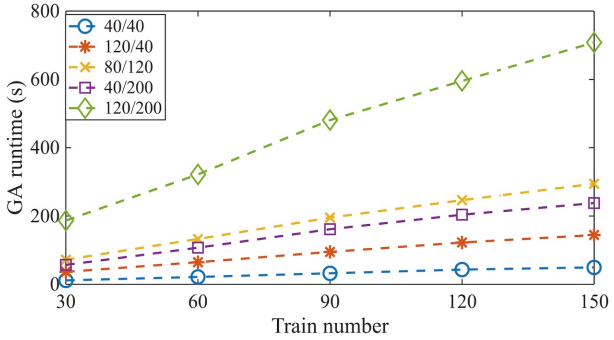

Fig. 4. Mean runtimes of our method under different parameters and train numbers

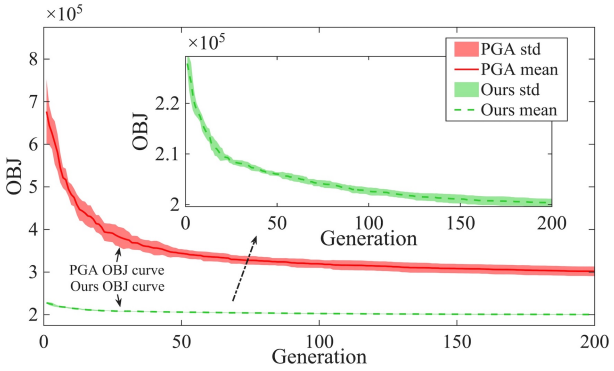

Fig. 5. Comparison of fitness curve during optimization

encountering the curse of dimensionality), and in experiments with practical-scale problems, the time required by our method is still acceptable. Moreover, as shown in Fig. 5, the useage of DFS decode method can lead to a faster and more stable convergence compared to PGA. So the benefits in convergence results can compensate for acceptable additional computing time.

5. CONCLUSION

In this paper, we schedule station operations, timetables and combination schemes of HHTs based on VC in an integrated way. We provided an IP model for the problem, and efficiently solved it with GA featured with a DFS decode method. In simulation experiments with real-world scenario settings, the effectiveness of our method is validated through comparison with results of GUROBI solver, and our method also demonstrates superior optimization outcomes compared to commonly used intelligent optimization methods and heuristics. However, we greedily combined trains when VC constraints are satisfied, and it may lead to reduction of the solution quality. In the future, more search-based methods can be introduced to further enhance decision-making related to VC.

ACKNOWLEDGEMENTS

This work was supported by the Research and Development Project of CRSC Research & Design Institute Group Co., Ltd.

REFERENCES

Amjad, M.K., Butt, S.I., Kousar, R., Ahmad, R., Agha, M.H., Faping, Z., Anjum, N., and Asgher, U. (2018). Recent Research Trends in Genetic Algorithm Based Flexible Job Shop Scheduling Problems. *Mathematical Problems in Engineering*, 2018(1), 9270802.

Chai, S., Yin, J., D'Ariano, A., Samà, M., and Tang, T. (2023). Scheduling of coupled train platoons for metro networks: a passenger demand-oriented approach. *Transportation Research Record*, 2677(2), 1671–1689.

Chaudhry, I.A. and Khan, A.A. (2016). A research survey: review of flexible job shop scheduling techniques. *International Transactions in Operational Research*, 23(3), 551–591.

Chen, C., Zhu, L., and Wang, X. (2022). An integrated train scheduling optimization approach for virtual coupling trains. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2182–2186.

Chen, C., Zhu, L., and Wang, X. (2024). Enhancing subway efficiency on y-shaped lines: a dynamic scheduling model for virtual coupling train control. *IEEE Transactions on Intelligent Transportation Systems*, 25(8), 10020–10034.

Davydov, Y., Kalikina, T., Plyaskin, A., and Keyno, M. (2017). The heavy haul train service on the eastern section of the Baikal-Amur mainline. *Procedia Engineering*, 187, 769–774.

Di Meo, C., Di Vaio, M., Flammini, F., Nardone, R., Santini, S., and Vittorini, V. (2020). ERTMS/ETCS virtual coupling: proof of concept and numerical analysis. *IEEE Transactions on Intelligent Transportation Systems*, 21(6), 2545–2556.

Ma, X., Zhou, M., Wang, H., Song, W., and Dong, H. (2024). Virtual-coupling-based timetable rescheduling for heavy-haul railways under disruptions. *IEEE Transactions on Computational Social Systems*, 11(5), 7045–7054.

Roshanaei, V., Azab, A., and ElMaraghy, H. (2013). Mathematical modelling and a meta-heuristic for flexible job shop scheduling. *International Journal of Production Research*, 51(20), 6247–6274.

Wang, Z. and Su, B. (2022). Integrated train timetable and rolling stock circulation plan scheduling with the virtual coupling technology for a Y-shaped metro line. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 1764–1769.

Wu, L., Ye, H., Dong, W., Jiang, M., Yu, X., and Xu, Z. (2024). An optimal scheduling method for heavy haul trains with virtual coupling technology. *International Journal of Rail Transportation*, 0(0), 1–26.

Wu, Q., Ge, X., Han, Q.L., and Liu, Y. (2023). Railway virtual coupling: a survey of emerging control techniques. *IEEE Transactions on Intelligent Vehicles*, 8(5), 3239–3255.

Zhou, H., Zhou, L., Guo, B., Bai, Z., Wang, Z., and Yang, L. (2021). A scheduling approach for the combination scheme and train timetable of a heavy-haul railway. *Mathematics*, 9(23), 3068.

Zhou, M., Liu, X., Wu, X., Song, H., and Dong, H. (2023). Timetable rescheduling for high-speed railways considering dynamical train groups in case of disturbances. In *Proceedings of 2021 5th Chinese Conference on Swarm Intelligence and Cooperative Control*, 1808–1819. Springer Nature, Singapore.