Proceedings of the 2nd International Conference on Mathematical Physics and Computational Simulation
DOI: 10.54254/2753-8818/39/20240601

Train dispatching program for high-speed railway station based on genetic algorithm

Haozhe Li

School of Mathematical Science, Tongji University, Shanghai, 200000, China

jimmyli@tongji.edu.cn

Abstract. In case of train delays, centralized traffic control system become disabled, and the workload of dispatchers increases dramatically. Based on genetic algorithm, the author designs a program to appropriately reschedule trains in terms of delays, minimizing the total delay time and changes of gate. The author transformed the initial problem to a compromised combinatorial optimization model, with total delay time, changes of gate and conflicting routes as objectives. The high weighting in conflicting routes ensures efficiency and high probability of obtaining a feasible solution. With discreate variants, the author designs special coding and evolving method suitable for this problem. Using a special treatment for conflicts and initializing chromosomes, the program can construct new timetable quickly given the scheduled timetable, predicted arrival time and order of trains (optional), which promotes the efficiency and security of dispatching in high-speed railway stations. The method was tested with a synthetic data of Shanghai-Kunming section of Hangzhou East Railway Station.

Keywords: Train dispatching, genetic algorithm, optimization.

1. Introduction

Large high-speed railway stations usually have a large scale of tracks and platforms and are often adjacent to the EMU depot. With a variety of types of trains such as origination, termination, entering and leaving the depot, and passing through, those stations possess a large number of conflicting and opposing train routes, increasing great difficulty in the safety and efficiency of train dispatching. Especially during peak hours of the station, once a train is delayed, it would have a consequential impact on the station's operational plans and adjacent operating trains. In severe cases, it may fully disrupt the existing timetable. When there is a large-scale train delays, the difficulty of manual adjustments become relatively high. Dispatchers need to comprehensively consider the nature and level of trains, as well as various factors such as water supply, waste disposal, track switching operations and EMU depot operations, thus difficult to make the optimal decision in a short period purely relying on manual experiences.

Kroon used Amsterdam Central Station and Utrecht Station in the Netherlands as examples to study the compilation of train routes [1]. However, the calculated results cannot effectively solve the problem. Subsequently, in 2001, the model was optimized, dividing the train routes into three types: arrival, departure, and origination-termination routes. The aim was to optimize the selection of preferred routes for trains [2].

DOI: 10.54254/2753-8818/39/20240601

Liu abstracted the arrangement of throat section turnouts as a network graph with multiple starting points and multiple junctions [3]. By establishing an optimization model and using computational programming methods, they conducted calculations on existing passenger stations. Conclusions were drawn regarding the reconstruction of throat sections of Liuzhou Station and Guiyang Station. Zhou optimized the issue of technical station turnout occupation [4]. Employing a method similar to Liu, after determining a fixed usage plan for station tracks, the author focused on selecting arrival and departure routes. Shi, with the optimization objective of ensuring the priority of high-level trains and maximizing the efficiency of station tracks, utilized modern simulated annealing algorithm to solve the problem [5].

In Li's work, a detailed analysis of the technical tasks and workflow of train stations was conducted [6]. With the background, an in-depth study of optimizing route selection for trains was carried out. Firstly, based on the description method of establishing station networks, a refined model covering various station elements was constructed. Secondly, train routes were described from both mathematical and formal perspectives, leading to the development of methods for generating train route tables. Thirdly, through an information model, the construction of station train operations and shunting operations was carried out. By avoiding train crossings on station routes based on completed planned operations, a train route selection model was established, resulting in optimal reductions in train travel time and delay rates. By analyzing the temporal correlations of route selection and route arrangement, the route selection problem was analogized to a 0-1 integer programming problem. Research results showed that the immune evolutionary algorithm could compute route selections satisfying constraints with good convergence. Addressing nonlinear models, new optimization algorithms were proposed by Long based on modern optimization algorithms [7]. With calculations it was demonstrated that the new optimization algorithms have higher solving speeds than modern optimization algorithms.

Chen conducted a comprehensive study on the route selection problem of train stations [8]. Firstly, comprehensive optimization was conducted for the utilization of station tracks and the selection of arrival-departure routes in a throat section. Secondly, the extension of arrival-departure routes from one end of the throat area to both ends of the station throat section, combined with arrival-departure lines, was termed as the "through-station route". Thirdly, aiming for the comprehensive optimization of through-station routes and shunting locomotive utilization, a 0-1 programming model for route selection was constructed, with compatibility between arrival-departure lines and turnouts as constraints, and the algorithm approach of Long was adopted to solve the model.

In Li's work, the focus was on the utilization of station tracks at large passenger stations during peak hours, considering the dual objectives of minimizing delay time for station tracks and maximizing passenger service quality [9]. The author proposed a multi-objective optimization model. Xia considered various uncertainties in station track utilization and established a stochastic planning model to improve efficiency and facilitate passengers [10].

With the contribution of previous scholars, the study focuses on researching computational methods for dealing with train delays at large high-speed stations. In case of train delays, the author aims to generate efficient auxiliary decision-making suggestions for track utilization adjustments at the station within a short time, thereby reducing the operational burden and enhancing the work efficiency of dispatchers.

2. Methodology

2.1. Problem statement

The information given for the problem, or the input data for the program are as follows: trains entering the station at the Shanghai-Kunming section of Hangzhou East Railway Station, and the order of trains entering or leaving each direction of the section. Information of trains includes the unique number of trains, scheduled arrival and departure time, predicted arrival time, direction of entry and exit, and the station track occupied by the train.

After processing the inputs, the program will provide the final decision of the arrival and departure time, and station track of trains, fulfilling the goal that the total delay and changes of gate are as small

DOI: 10.54254/2753-8818/39/20240601

as possible. The author makes several assumptions to reduce problem complexity while maintaining the effect of analysis. The first assumption is for the minimum dwell time and available tracks as follows (Table 1, 2).

Table 1. Assumption of minimum dwell time and available track.

Assignment type	minimum dwell time	Available tracks
Station stops	2min	All
Water supply	5min	Except XIXG, XXG
Suction of dirt	10min	17G, 18G, 21G, 22G
Reversing/ starting and ending	15min	Except XIXG, XXG

The second assumption is for train intervals.

Table 2. Assumption of train intervals.

Type	Interval
Occupying the same route for Shanghai or Kunming	3min
Occupying the same EMU depot route	5min
Occupying the same platform	3min

The third assumption is that the route selection of each train from each direction to each platform is unique. According to analysis for the topology structure of this case, which is the Shanghai-Kunming section of Hangzhou East Railway Station, for any situation with parallel routes, there is always a route that minimizes its conflict with all other routes, so the author selects a unique route for each train. If this model is adopted to other stations it required to satisfy this assumption.

Moreover, the writer assumes that the actual departure time cannot be earlier than the scheduled departure time as a requirement for transportation organization. And the writer intends to research on discreate variables, so the given arrival and departure time is only accurate to the minute.

2.2. Symbol table and index description

First, the number of trains are indexed $i \in 1,2,\dots n$, where n is the total number of trains. Second, the writer indexed tracks and routes uniformly, where the track index are their original numbers from 14 to 25, and the route is indexed in a certain order starting from 31 to 87. About the direction d of entering and leaving the station, the meaning of each d is (Table 3),

Table 3. The meaning of numbering d.

d	direction of entry and exit
1	Shanghai end of Shanghai-Kunming Expressway upward line
2	Shanghai end of Shanghai-Kunming Expressway downward line
3	Kunming end of Shanghai-Kunming Expressway upward line
4	Kunming end of Shanghai-Kunming Expressway downward line
5	EMU depot line A
6	EMU depot line B

Last the assignment type $type$ (Table 4) is as follows.

Table 4. The meaning of $type$.

$type$	Assignment type
1	Station stops
2	Water supply
3	Suction of dirt
4	Reversing/ Starting and ending

DOI: 10.54254/2753-8818/39/20240601

2.3. Problem analysis

The author has transformed the train dispatching problem in the Shanghai-Kunming section of Hangzhou East Railway Station into a compromised optimization model and has solved it using a modified genetic algorithm.

2.3.1. Variants. The variants of this genetic algorithms are as follows: $\overrightarrow{t_1} = (t_{1,1}, t_{1,2}, \ldots, t_{1,n})$, which represents the actual arrival time vector of all trains. $\overrightarrow{t_2} = (t_{2,1}, t_{2,2}, \ldots, t_{2,n})$ represents the actual departure time vector of all trains. $\vec{r} = (r_1, r_2, \ldots, r_n)$ represents the station track number vector of all trains. The total decision vector is synthesized in the following way: $x_i = (t_{1,i}, t_{2,i}, r_i)$ and $\vec{V} = (x_1, x_2, \ldots, x_n)$, where, $t_{1,i} = \vec{V}(3i-2), t_{2,i} = \vec{V}(3i-1), r_i = \vec{V}(3i)$ and $\vec{V}(t)$ represents the t-th component of vector $\vec{V}$.

2.3.2. Constraints. The first constraint is the ingress and egress direction order constraint. Since high-speed railway stations may do not have the right to change the order of trains thus require a higher-level dispatcher to decide the order of trains entering and leaving the station, the author considers this as an optional constraint.

Suppose direction d has an order $i_1, i_2, \ldots, i_l$, indicating that train i_k needs to pass this direction ahead of traini_{k+1}. Let δ_k be the access symbol of train i_k ($\delta_k = 1$ as enter, $\delta_k = 2$ as exit), then the direction order constraint is:

$$t_{\delta_{k+1}, i_{k+1}} - t_{\delta_k, i_k} \geq \Delta t_d^{ocp} \tag{1}$$

Where Δt_d^{ocp} indicates the minimum time interval of direction d, which is:

$$\Delta t_d^{ocp} = \begin{cases} 3min, & d = 1,2,3,4 \\ 5min, & d = 5,6 \end{cases} \tag{2}$$

The second constraint is the arrival time constraint. The actual arrival time of the train must not be earlier than the predicted arrival time, that is:

$$t_{1,i} \geq t_i^{pre} \tag{3}$$

The third constraint is the departure time constraint. The actual departure time of the train can neither be earlier than the scheduled departure time nor the theoretical earliest departure time. The theoretical earliest departure time means, the time train finishes its assignment after it arrives the station.

$$t_{2,i} \geq max\{ t_{2,i}^{sch}, t_{1,i} + \Delta t_{type_i}^{work} \} \tag{4}$$

The fourth constraint is the station track constraint.

Table 5. Available station tracks based on assignment type.

$type_i$	Selectable station track r
1	14G, 15G, …,25G
2	14G, 15G, …,18G, 21G, …,25G
3	17G, 18G, 21G, 22G
4	14G, 15G, …,18G, 21G, …,25G

At the same time, the ingress and egress direction of each train also affects the station tracks it can occupy. The final available station tracks for the train are the intersection of both constraints (Table 5).

Before moving to next section, there are some explanations for why conflicting routes are not considered as a constraint. To ensure the effectiveness of the algorithm and prevent the feasible region from possessing a bad topological structure, the author places the number of route conflicts in the optimization function and assign it a large weight. The purpose enabling the algorithm to quickly obtain an acceptable solution that is unlikely to contain conflicts.

DOI: 10.54254/2753-8818/39/20240601

2.3.3. Objective function. Denote $f(\vec{V}) = \lambda_1 Z_1 + \lambda_2 Z_2 + \lambda_3 Z_3$, where $\lambda_1, \lambda_2, \lambda_3$ are the weighting coefficient, Z_1 is the total delay minutes, Z_2 is the changes of track, and Z_3 is number of conflicting routes. In practical application, the author takes $\lambda_1 = 1$, $\lambda_2 = 10$, and $\lambda_3 = 100$. Ratio $\frac{\lambda_2}{\lambda_1}$ can be viewed that in practice, the impact of changing one station track approximates the impact of $\frac{\lambda_2}{\lambda_1} = 10$ minutes of total delay. It can be deduced that,

$$Z_1 = \sum_{i=1}^{n} \left(t_{1,i} - t_{1,i}^{sch}\right) + \sum_{i=1}^{n} \left(t_{2,i} - t_{2,i}^{sch}\right) \tag{5}$$

and

$$Z_2 = \sum_{i=1}^{n} \left(r_i \neq r_i^{sch}\right) \tag{6}$$

where $\left(r_i \neq r_i^{sch}\right)$ represents the Boolean value of the difference between the actual station track and the scheduled station track of train i, that is,

$$\left(r_i \neq r_i^{sch}\right) = \begin{cases} 1, r_i \neq r_i^{sch} \\ 0, r_i = r_i^{sch} \end{cases} \tag{7}$$

To calculate $\boldsymbol{Z_3}$, the number of conflict routes, let $conf_{r_1,r_2}$ indicate the conflicting Boolean value of r_1, r_2, where $conf_{r_1,r_2} = 1$ when r_1, r_2 conflict and $conf_{r_1,r_2} = 0$ when they do not. Notice that $conf_{r,r} = 1$.

Define occupation function $ocp_{r,t}$ indicating the number of trains occupying route or track r during $[t, t+1)$. Then the program marks the time each train occupies the route or track in $ocp_{r,t}$. Then it can be deduced that,

$$Z_3 = \sum_{r_1,r_2,t} ocp_{r_1,t} ocp_{r_2,t} conf_{r_1,r_2} - \sum_{r,t} min\{ocp_{r,t}, 1\} \tag{8}$$

The first sum represents the number of conflicts for all routes and tracks, but it includes an overcount of $r_1 = r_2$, which means self-conflict. Because any route or track can afford occupation up to 1, the second sum needs to subtract the maximum of 1 occupation.

2.4. Algorithm design

2.4.1. Initializing chromosomes. The goal is to generate random vectors satisfying the constraints in 2.3.2. as quickly. The initial idea was to first randomly generate random vectors in a large enough hyperrectangle and repeat the process until the vectors satisfy all constraints. But after testing the writer found that its efficiency is far from satisfactory, costing 10 times longer time than other steps. Then the author transformed this step into a graph theory problem and solved it with topological sorting, which was proved effective.

It can be observed that any time constraint in 2.3.2. have a format of either $t_i \geq a_i$ or $t_i - t_j \geq a_{i,j}$. The former can be transformed to the latter by adding a virtual time t_0 which is set to *0*, thus $t_i \geq a_i$ becomes $t_i - t_0 \geq a_{i,0}$. All the time variants t_i and the virtual time t_0 can be viewed as nodes of a graph, and constraint $t_i - t_j \geq a_{i,j}$ represents an edge from t_j to t_i with weight $a_{i,j}$.

Since the existence of a solution is guaranteed by the problem itself, there is no ring in the graph thus a topological ordering exists. One can observe that the only node with indegree 0 is the virtual time node t_0, so t_0 must be ordered first. Initialize all nodes with value 0 (noticing that $t_0 = 0$), and visit all the nodes and their outgoing edges in the topological ordering. For each edge from t_j to t_i, set the value of

DOI: 10.54254/2753-8818/39/20240601

t_i to the maximum of the current value and $t_j + a_{i,j}$. Then the smallest vector $\vec{t}$ satisfying all constraints is obtained.

To get random vectors to form the population of chromosomes, simply enlarge the weight on all edges randomly and execute the topological sorting method. After N repeated processes the initial chromosomes are generated.

2.4.2. Crossover. Define the parameter P_c the probability of crossover operation. This probability indicates that there are expected $P_c \cdot N$ chromosomes subjected to crossover operations.

To determine the parents of cross operation, repeat the following process for i in $1 \sim N$: generate a random number from [0,1] denoted ran. If $ran < P_c$, select $\vec{V_i}$ as a parent. Randomly permute the selected parents and combine them into the following:

$$\left(\overrightarrow{V_1'}, \overrightarrow{V_2'}\right), \left(\overrightarrow{V_3'}, \overrightarrow{V_4'}\right), \ldots, \left(\overrightarrow{V_{s-1}'}, \overrightarrow{V_s'}\right) \tag{9}$$

If the number of vectors is odd, the last one will be discarded. For each pair $\left(\overrightarrow{V_k'}, \overrightarrow{V_{k+1}'}\right)$, generate a random number t indicating the location of crossover. Assuming $\overrightarrow{V_k'} = (x_1, x_2, \ldots, x_n)$, $\overrightarrow{V_{k+1}'} = (y_1, y_2, \ldots, y_n)$. After crossover, the two chromosomes become $\overrightarrow{V_k'} = (x_1, x_2, \ldots, x_t, y_{t+1}, \ldots, y_n)$ and $\overrightarrow{V_{k+1}'} = (y_1, y_2, \ldots, y_t, x_{t+1}, \ldots, x_n)$. Intuitively, the two chromosomes exchanged strategies for the first t trains. If either chromosome is illegal after crossover, the crossover operation is abandoned.

2.4.3. Variation. Define the parameter P_{m_t} the probability of mutation of time. This probability indicates that there are expected $P_{m\_t} \cdot N$ chromosomes subjected to variation operations.

First, determine the parents of the time mutation operation like the crossover operation. For each selected mutant parent $\vec{V}$, denote its variants of time as vector $\vec{t}$. For this mutation operation, Give the upper bound of time variance M and randomly select the mutation position $k, k \in [1,2n]$. Generate a random integer from $[-M, M]$ denoted $\Delta\vec{t}$ and add it to the k-th dimension of $\vec{t}$, and call the modified vector $\vec{t}\,\vec{t}\,'$. If $\vec{t}\,'$ is illegal, let M be its half and mutate again until a feasible time vector is obtained (ultimately $M < 1$ and $\vec{t}\,' = \vec{t}$ must be legal).

To mutate station track, define the parameter P_{m_r}. Select parents as above and randomly select the mutation position $k, k \in [1, n]$. For the k-th train, randomly select one of its available station tracks.

2.4.4. Evaluation and selection. The author uses an order-based evaluation function to evaluate and select chromosomes. For chromosome population $\overrightarrow{V_1}, \overrightarrow{V_2}, \ldots, \overrightarrow{V_N}$, calculate the objective function of each chromosome $f\left(\overrightarrow{V_i}\right)$. Then, sort the chromosomes by objective function value from smallest to largest. Denote p_i the position of $\overrightarrow{V_i}$ after sorting. Thus, the evaluation function is defined as,

$$eval\left(\overrightarrow{V_i}\right) = a(1-a)^{p_i - 1}, i = 1,2, \ldots, N \tag{10}$$

where $a \in (0,1)$ is a predetermined parameter, and the author takes $a = 0.05$. Then, select chromosomes based on this evaluation function. For each chromosome, calculate the cumulative probability:

$$q_i = \sum_{j=1}^{i} eval\left(\overrightarrow{V_j}\right), i = 1,2, \ldots, N \tag{11}$$

DOI: 10.54254/2753-8818/39/20240601

Generate a random number ran from interval $(0, q_N]$. If $ran \in (q_{i-1}, q_i]$, select the chromosome $\vec{V_i}$. Repeat the procedure N times to get N newly selected chromosomes.

2.4.5. Output. After each evaluation, update the chromosome $\overrightarrow{V^{best}}$ with the minimum objective function at all times. Finally, the program outputs the optimal solution during the execution process.

3. Results and discussion

3.1. Case testing

The author created a running schedule for 21 trains in the Shanghai-Kunming section of Hangzhou East Railway Station in one hour and set delays randomly. After several executions of the program, the optimal solution for this case is 149 minutes of delay with 1 change of gate. Both scheduled and adjusted train operation are drawn below, where each bar represents the time interval a train occupies a station track or direction (Figure 1, 2).

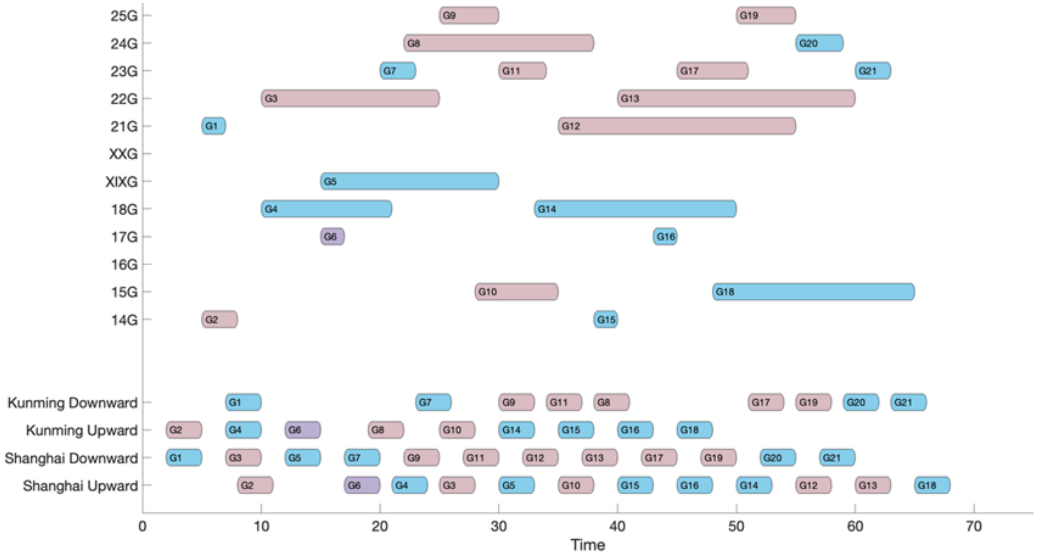

Figure 1. Scheduled train operation diagram.

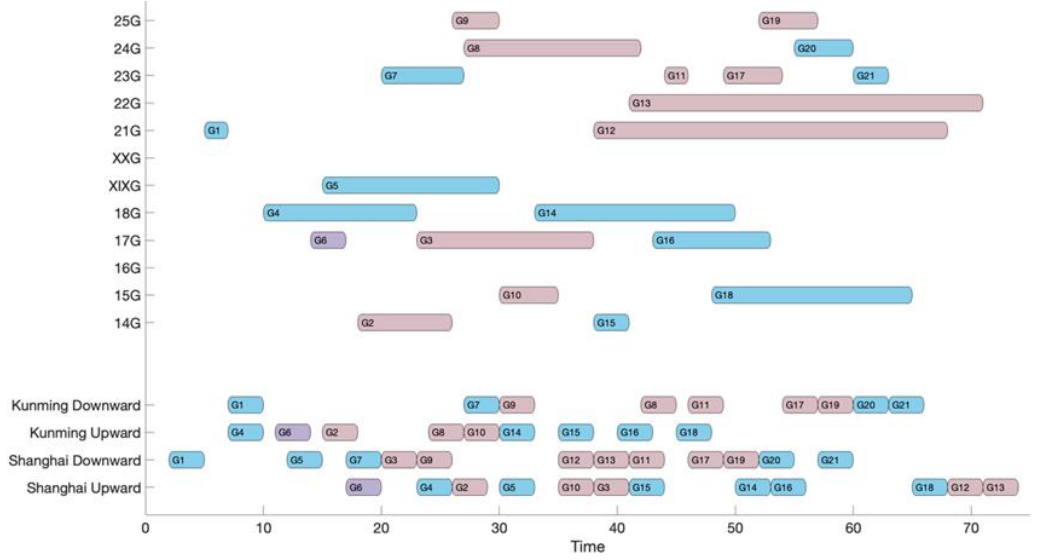

Figure 2. Adjusted train operation diagram.

DOI: 10.54254/2753-8818/39/20240601

The author uses different color to represent different status of trains. Blue trains are on their schedule, purple trains are ahead of schedule, and red trains are delayed. The program slightly adjusted the dwell time of trains to minimize overall delay time, and the station track of G3 is changed from 22G to 17G to avoid conflicts with G12 and G13 to cause widespread delay. Intuitively the program is relatively effective and further analysis of the algorithm is conducted below.

3.2. Algorithm analysis

3.2.1. Overview. In terms of solving efficiency, genetic algorithms use natural selection and genetic mechanisms to find the optimal solution, which can usually find relatively good solutions in a relatively short period of time. In addition, the convergence speed of the algorithm is also affected by parameter settings, such as population size, crossover rate, and mutation rate. By reasonably adjusting and optimizing these parameters, the solving efficiency of the algorithm was improved. In testing the author adjusted P_c, P_{m_t} and P_{m_r}, and the size of the chromosome population is reduced to shorten the time each loop without losing much accuracy.

In terms of solution quality, genetic algorithm can find relatively good solutions, but in most cases, it may not be able to find the optimal solution or even feasible solution. To increase the solving accuracy, the writer tested the stability of this program and calculated the required execution times for high probability of success.

3.2.2. Efficiency analysis. The time complexity and space complexity of computer program are important indicators for measuring the efficiency of the algorithm. Time complexity focuses on the change in the running time as the problem size changes, while space complexity focuses on the storage space.

For to the genetic algorithm the author adopts, the complexity of calculating the objective function is $O(R^3)$, where R represents the sum of all routes, ports, and tracks, in the test case $R = 90$. The determining factor is calculating conflicting routes $\sum_{r_1,r_2,t} ocp_{r_1,t} ocp_{r_2,t} conf_{r_1,r_2}$ in Z_3. The time complexity of crossover and mutation is both $O(N)$. Therefore, the time complexity of performing the genetic algorithm is $O((R^3 + N)G)$.

The space complexity mainly depends on factors such as population size and encoding method. Let the population size be N, the space complexity of storing chromosomes is $O(N)$. When calculating the objective function, the program constructs an occupation matrix ocp, whose space occupation is $O(NR)$. Since the chromosomes and occupation matrix is updated in every loop, the total space complexity is independent of the number of evolutions, which is $O(NR)$.

3.2.3. Stability analysis. The stability of genetic algorithm refers to the ability of the algorithm to converge steadily to the optimal or feasible solution. To evaluate the stability of the program, the author used the method of repeated experiments. The author executed the program 200 times for the above case and analyzed the results and running time. For each execution of the genetic algorithm, if the optimal objective function value did not change after 50 evolutions, the execution was terminated.

The 200 executions of the program took 791.26 seconds, with an average of 3.96 seconds per execution. Sorting all the results outputted, the writer obtained the following plot of the objective function values for the 200 executions:

DOI: 10.54254/2753-8818/39/20240601

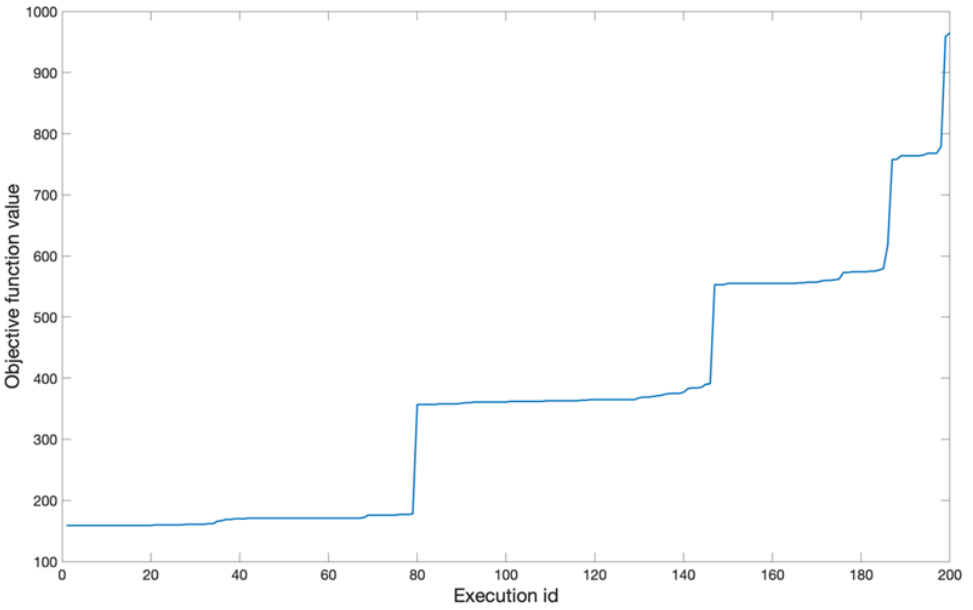

Figure 3. The graph of objective function values running 200 times.

From the figure 3, the running results have roughly 5 "ladders", and only on the first ladder the objective function values are below 200, which is feasible. Analyzing the data in detail, the frequency of obtaining a feasible solution ($Z_3 = 0$) is 39.5%. The frequency of optimal solution 159 is 10%, and the frequency of the objective function value below 170 is 20.5%.

Suppose that obtaining feasible solution obeys a binomial distribution with success rate $p = 39.5\%$. If the success rate reaches 99.5% then it is necessary to execute the genetic algorithm at least 11 times, which takes about 43.52 seconds. The predicted running time is acceptable for practical situation.

4. Conclusion

The problem aroused from the fact that centralized traffic control system is not capable of dispatching trains in terms of large-scale train delays. The program uses scheduled timetable, predicted arrival time and order of trains (optional) as inputs, and outputs an adjusted timetable minimizing the total delay time and changes of gate. The author innovatively treats the conflicting routes as an optimization goal rather than a constraint to increase the computational efficiency, and transformed all linear constraints to a graph to accelerate initializing chromosomes by topological sorting.

With the arrival time, departure time and station track as discrete variants of genetic algorithm, the author modified the common operations of initialization, crossover, and mutation of genetic algorithm. Intuitively, crossover operation switches the strategies of the first half trains for each pair of parents, and mutation operation changes one time variant or one station track variant for selected chromosomes. Initialization operation requires to transform all constraints for the difference between two time variants to a graph and do topological sorting. Based on the hypothesis of unique train routes, the program pretreats a conflict matrix from each direction to each station track. Then it can calculate the conflicts in each chromosome by matrix multiplication. Combining this quality with the total delay time and changes in gate with different weights, the program can calculate the evaluation function and select better-performed chromosomes.

The program is tested effective in the case the author synthetized. Then this paper further analyzed its efficiency and stability. By setting the weighting coefficient of conflicting routes relatively high, the program has a success rate of nearly *40%*. Calculating by binomial distribution it is obtained that to get a success rate of *99.5%* approximately requires a running time of 40 seconds. Considering the test case lasts for about one hour, the efficiency and accuracy of the program is generally acceptable.

DOI: 10.54254/2753-8818/39/20240601

References

[1] Zwaneveld P J, et al. 1996 Routing Trains Through Railway Stations: Model Formulation and Algorithms. Transportation Science, 30(1), 181-194.
[2] Zwaneveld P J, Kroon L G and Hoesel S R M 2001 Routing Trains Through a Railway Station Based on a Node Packing Model. European Journal of Operational Research, 128(1), 14-33.
[3] Liu L, Wang N and Du W 2002 Research on Network Optimization Model and Algorithm for Throat Passing Capacity of Stations. Journal of Railways, 1-5.
[4] Zhou Z L, You B and Li X T 2002 A model and algorithm for the occupancy arrangement of switch groups in the throat area of railway technical stations. Journal of Sichuan Institute of Technology, 70-72.
[5] Shi F, Chen Y, Qin J and Zhou W L 2009 Comprehensive optimization of the application of arrival and departure lines and the arrangement of arrival and departure routes in railway passenger stations. China Railway Science, 108-113.
[6] Li Y H 2007 Research on Immune Evolution Algorithm for Railway Station Route Selection. Beijing Jiaotong University.
[7] Long J C, Gao Z Y, Ma J J and Li K P 2007 Research on Optimization Model and Solution Algorithm for Railway Station Route Selection. Journal of Railways, 7-14.
[8] Chen Y 2010 Optimization of Train Passing Path and Adjustment Operation at Passenger Stations. Changsha: Central South University.
[9] Li Y H 2010 Research on Optimization Scheme for the Operation of Arrivals and Departures at Large Passenger Stations during Peak Hours. Beijing Jiaotong University.
[10] Xia M, Zhou L S, Le Y X, Zhou Y F and Zhou Y 2010 Research on Random Chance Constraint Model and Algorithm for the Application of Arrival and Departure Lines in Passenger Dedicated Stations. Logistics Technology, 8, 54-58.