# Genetic Algorithms Implement in Railway Management Information System

Jia Li-Min[1] and Meng Xue-Lei[2]
[1]*State Key Laboratory of Rail Traffic Control and Safety, Beijing Jiaotong University, Beijing,*
[2]*School of Transport and Traffic, Lanzhou Jiaotong University, Lanzhou, China*

## 1. Introduction

The railway system is a complex system and accordingly, the processes of railway operations are very complicated. The essence of railway operations is to meet the demand assigned to railways through the optimization of usage of the railway transportation specific resources. So problem-solving oriented optimization algorithms or techniques form the basis, especially in the era when information technologies prevail, of modern railway operation, in which the most typical cases are, not limited to, as follows.

- Railway management information systems design
- Transportation resources allocation
- Traffic control through dynamical transportation resources re-allocation

These problems feature large scale, combinatorial complexity and nonlinearity etc and need optimization algorithms with searching efficiency, scale insensitivity, embedded heuristics and certainty of solution-finding. Genetic Algorithms (GA) have been proved suitable for the above mentioned railway application oriented problems solving.

The chapter consists of two parts. The first part will cover some recent development of and improvement to the generic GA. And the second part will outline several typical applications of GAs to solving optimization problems arising from railway management information system design, transportation resources allocation and traffic control for railway operation.

## 2. Improvement on GA

GA is the most popular type of EA. One seeks the solution of a problem in the form of strings of numbers (traditionally binary, although the best representations are usually those that reflect something about the problem being solved), by applying operators such as recombination and mutation (sometimes one, sometimes both) (Bäck 1996). The most important development on GA is the improvement on its efficiency and precision. And the other achievement we got at hand is that the implement of GA in optimization algorithm, which is not only used to solve single objective problem, but also used to solve multi-objective problem. We will introduce this improvement to the GA from our research group. We mainly introduce a variation of GA: The new kind of fast global convergent multi-objective genetic algorithm (Meng 2005).

The key step of genetic algorithm is to select the best chromosomes of each generation to generate the next generation, and the sorting methods of chromosomes are the most important, which determines the selecting results. So we focus on the sorting methods for selection and propose a new kind of sorting method---Improved Non-dominated Sorting method, constructing the basis of the fast global convergent multi-objective genetic algorithm.

### 2.1 Improved non-dominated sorting method

First, we calculate the number of dominating individuals for individual $i$ --- $n_1$; then, we propose the sets of dominated individuals by individual *i*--- $S_1$. The steps are as follows.

**Step 1.** To find the individuals which satisfy the equation $n_1 = 0$ and store them into $F_1$.

**Step 2.** To propose $S_i$. For every individual *j* in $S_i$, there are $n_j$-1 dominating individuals. If $n_j$-1=0, then store individual *j* into another set $H$.

**Step 3.** Set $F_1$ to be the first dominating individuals set. Continue these steps to deal with $H$ till each individual is graded.

The time complexity of the algorithm is $O\left(mN^2\right)$. $m$ is the number of the objects and *N* is the population size of the chromosomes. *P* is the population, *F* is the non-dominating individuals set (It is the non-inferior solutions set). The algorithm is as follows.

Algorithm

```
For every $p \in P$
 For every $q \in P$
  if $p \prec q$, then $S_p = S_p \cup \{q\}$
  if $p \prec q$, then $n_p = n_p + 1$
  if $n_p = 0$, then $F_1 = F_1 \cup \{p\}$
 $i = 1$
while $F_1 \neq \Phi$
   $H = \Phi$
   for every $p \in F_1$
    for every $q \in S_p$
     $n_q = n_q = 1$
     if $n_q = 0$ then $H = H \cup \{q\}$
   $i = i + 1$
   $F_1 = H$
```

### 2.2 Dynamic virtual adaptability strategy

The selection probability of the individual is usually based on the adaptability according to a direct ratio. As a result, the value of the virtual adaptability can affect the selection probability of the individuals. If the difference between the adaptabilities of different groups is obvious, the group which has the bigger adaptability is selected and the individuals in the group will get the dominating positions, which is an obstacle to maintain the diversity of the individuals. We propose a dynamic virtual adaptability strategy to avoid the effect.

As discussed above, the virtual adaptability functions $f_1$ of individual $i$ is as follows, with its grade $Level$.

$$f_i = \frac{1}{A\left(\frac{Maxgen - gen}{Maxgen}\right) + B^{*}Level} \tag{1}$$

In the function definition, $Maxgen$ is the maximal iteration times, $gen$ is the current generation. $Level$ is the grade of the individual *i*. A and B are the two adjustable parameters, which can be used to adjust the change speed of the difference between different grades according to the grades and the algebra. Take it for granted that the individuals can be divided into 50 grades and the changing graph can be attained according to equation (1), as shown in Figure 1. The smaller the generation is, the smaller the differences between the adaptabilities of different groups are. The difference becomes bigger and bigger when the iteration continues. And the higher the grade of the individual belongs to, the bigger adaptability the individual will get. Then, the probability of being selected is bigger which is propitious to make the process of searching the optimal solution convergent.

### 2.3 Maintenance of the diversity of individuals

The local convergence can often be found when solving the multi-objective problem, so something must be done to maintain the diversity of the individuals. To share the adaptability is a method. The basic idea is to take the adaptability as the public resources and to adjust the adaptability according to the number of the similar individuals and the similarity of an individual. This can lead to two results. One is to form a niche among the similar individuals. The other is to control the number of the individuals generated by the similar individuals which can help to maintain the diversity the individuals. The shared function $Sh(\bullet)$ is introduced to measure the similarity between the individuals, which is defined as follows, shown in equation (2).

$$Sh\left(d_{ij}\right) = \left\{\begin{array}{l} 1-\left(\frac{d_j}{\sigma_{share}}\right)^{\alpha}, \text{if } d_{ij} < \sigma_{share} \\ 0, \text{else} \\ \\ \end{array}\right\} \tag{2}$$

$\alpha$ is a constant, $\sigma_{share}$ is the radii of the niche which can be estimated according to the expected minimal dividing degree of the individuals. $d_{ij}$ is the distance between $i$ and $j$.

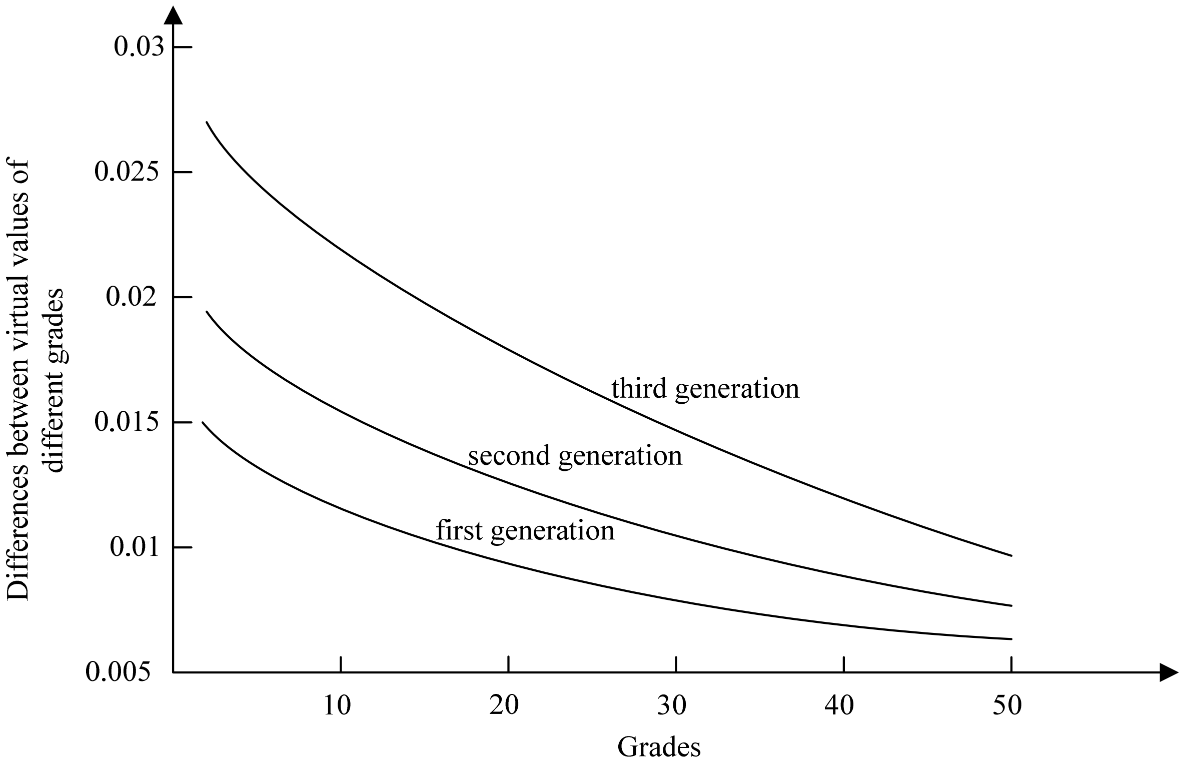


Fig. 1. Differences between virtual values of different grades

When the adaptability sharing function is defined, the adaptability of the individual $i \in P$ is the quotient of the adaptability before sharing operation and the number of niche --- $m_i$ .

$$f_i = \frac{f_i}{m_i} \tag{3}$$

$$m_i = \sum_{j=i}^{pop\_size} Sh\left(d_{ij}\right) \tag{4}$$

According the different definitions of $d_{ij}$, the sharing methods can be divided into two kinds: the share in the objectives and the share in the solutions space. Here we introduce the first kind of method. $d_j$ is defined as equation (5).

$$d_{ij} = d\left(f_i^{'}, f_j^{'}\right) \tag{5}$$

### 2.4 Elitist strategy

To hire the elitist strategy can greatly improve the convergence of the Pareto solution set. There are many kinds of methods for selection of elitists. Usually it can be selected from the non-inferior solutions set in which the solutions have been sorted. The number of the individuals to be selected is decided according to the problem condition. The methods to realize the elitist strategy are as follows. One is to add the elitist individuals into the current generation and delete some of the individuals which are worst. The other is to add the elitist individuals into an external set $\overline{P}$ , and select some of the individuals from $\overline{P}$ and replace some of the individuals of each generation. Then some of the elitist individuals of the current generation are selected to be added into $\overline{P}$ . And some methods are designed to control the number of the elements in $\overline{P}$ and the number of the similar individuals in $\overline{P}$ .

We design a new method here. Each individual in the first grade after grading is added into the elitists pool (the external set $\overline{P}$ ). And the new generation $P^{'}$ is generated after selection, crossover, and mutation. Each individual (the number is suspended to be $K$ ) in elitists pool is added into $P^{'}$ directly. And to maintain the population size $N, N-K$ non-elitist individuals are deleted randomly.

### 2.5 Algorithm flow

The steps of the algorithm are as follows.

**Step 1.** The initial population $P(0)$ is generated through fake random method.

**Step 2.** To grade the individuals.

**Step 3.** To refresh the individuals in elitists pool. To store the individuals of the first grade in the elitists pool.

**Step 4.** To calculate the virtual adaptability according to equation (1).

**Step 5.** To apply the share strategy in objectives functions space according to equation (2) ~ equation (5).

**Step 6.** Selection: To select the individuals according to the adaptability ratio method (roulette).

**Step 7.** To crossover.

**Step 8.** To mutate.

**Step 9.** To generate the new generation $P^{'}$ .

**Step 10.** To apply the elitist strategy and add the individuals in elitists pool into the new generation $P^{'}$ and delete the non-elitist individuals in $P^{'}$ randomly.

**Step 11.** To adjust the size of the new generation. Set the size to be $N$ . The new generation is $P$ .

**Step 12.** To check if the ceasing condition is satisfied. If no, go to step 2; else, cease.

The flow chart is shown in Figure 2.

## 3. Railway management information systems design

The most important work in railway management information system design is to design the physical structure. The essence of physical structure optimization is to allocate the physical resources of railway in order to realize logical structure, minimizing the costs and time.

The optimization objectives include customer satisfactory degree, response time, costs and so on. The constraints are reliability, safety and etc. So we can see the physical structure optimization problem features objective variety, combinatorial complexity and objective index nonlinearity etc.

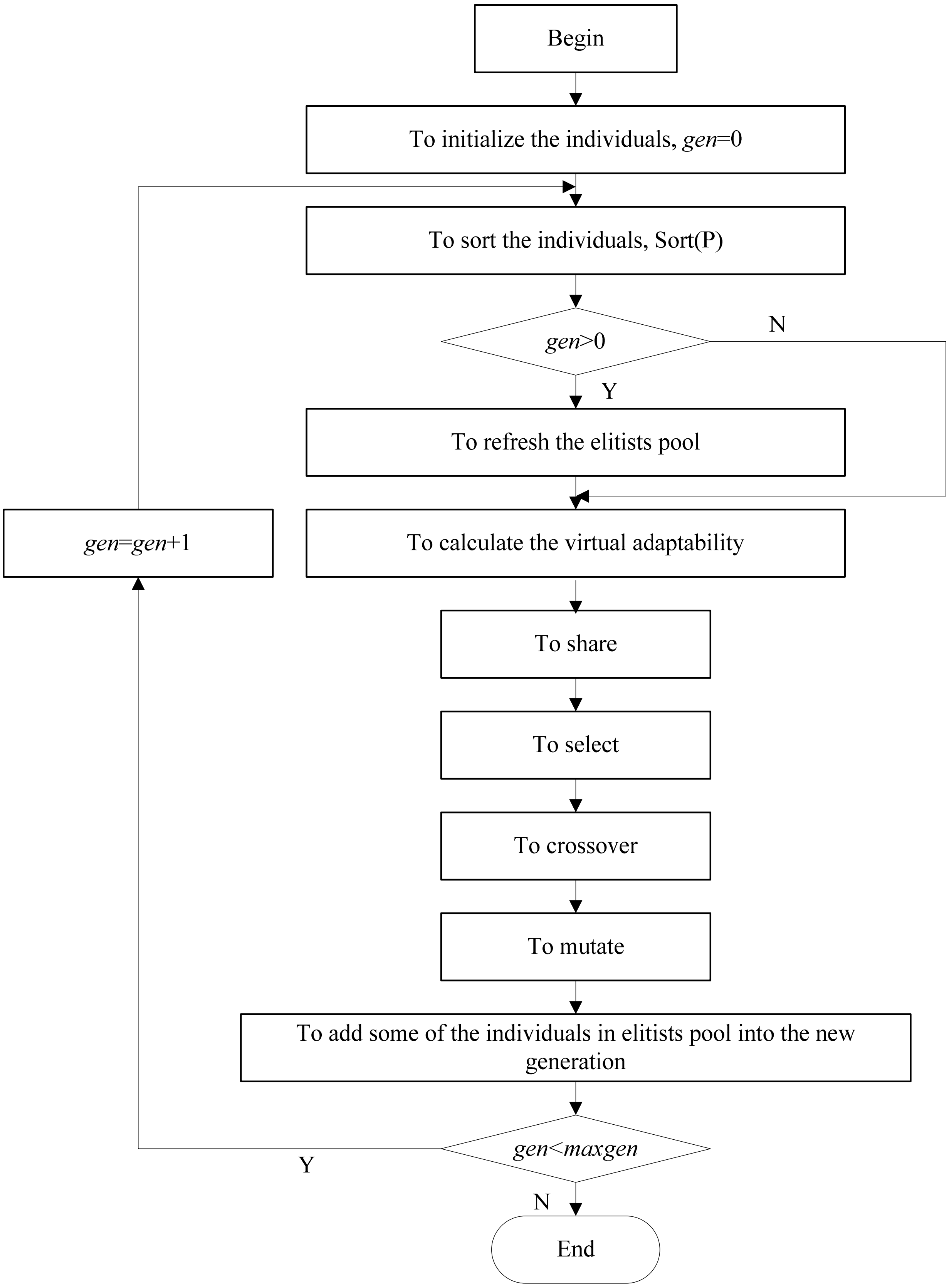

Begin
To initialize the individuals, *gen*=0
To sort the individuals, Sort(P)
*gen*>0
N
Y
To refresh the elitists pool
*gen*=*gen*+1
To calculate the virtual adaptability
To share
To select
To crossover
To mutate
To add some of the individuals in elitists pool into the new generation
*gen*<*maxgen*
Y
N
End

Fig. 2. Flow chart of the new genetic algorithm

A typical application of GA in railway management information system design is the physical structure optimization of emergency succoring system. The key points of implementing GA in the physical structure optimization of emergency succoring system are as follows.

(1) To describe the allocation plan of the physical equipments, the chromosome will be designed to be formed with two integers. The first integer denotes the allocation pattern of each function unit in the sub-systems while another describes the allocation pattern on the equipments of logical procedure.

(2) The objectives, the computing ability and operation cost can both be described with the chromosomes codes, forming the adaptive function formulas.

(3)The optimizing process is to find the optimal adaptive function value through all the iterations, with the chromosomes being selected, crossed over and mutated, deciding the optimal chromosome and the generation it belongs to.

A typical application of GA in railway management information system design is the physical structure optimization of emergency succoring system. Here we give the detailed implementing process of GA in physical structure optimization of emergency succoring system through a computing case study (Meng 2005).

### 3.1 The description of the physical structure optimization of emergency succoring system problem

The system is divided into 5 physical sub-systems, which are marked with $v_i\left(i=1,2,3,4,5\right)$ .The goal is to optimize the calculating ability of the whole system$\left(f_i\left(x\right)\right)$ and minimize the operating cost of the system $\left(f_2\left(x\right)\right)$, which can be described as follows.

$$\min\left(f_1\left(x\right),f_2\left(x\right)\right) \tag{6}$$

The Data Flow Diagram (DFD) of emergency succoring system is shown in Figure 3. The logic processes of the emergency succoring system can be abstracted into 18 function units. They are {1},{2,3},{8,9},{4,5,6,7},{10},{11,12,13},{15,16,17},{19,20,21},{23,24,25}, {27},{28,29,30},{31},{32},{33},{34},{35},{36,37},{38,39}. The meta-services can be described with the nodes lists as follows.

### 3.2 The computing case

(1) Coding approach

To distribute the meta-services on the physical sub-systems and optimize the allocation of the processes on the equipments, the genetic codes are designed to be planar. The chromosome consists of two integers shown in Figure 4. The first denotes the distributing modes of the function units on the system. For example, $S$ denotes the number of the sub-system that the function unit $F_i$ belongs to and $n$ is the total number of the function units. The second describes the allocating modes of the logic processes on the system equipments. For instance, $ID_i^k$ denotes the equipment number of process $P_i^k$ allocated on physical system $v_i$ . $m$ is the total number of logic processes in $v_i$ .

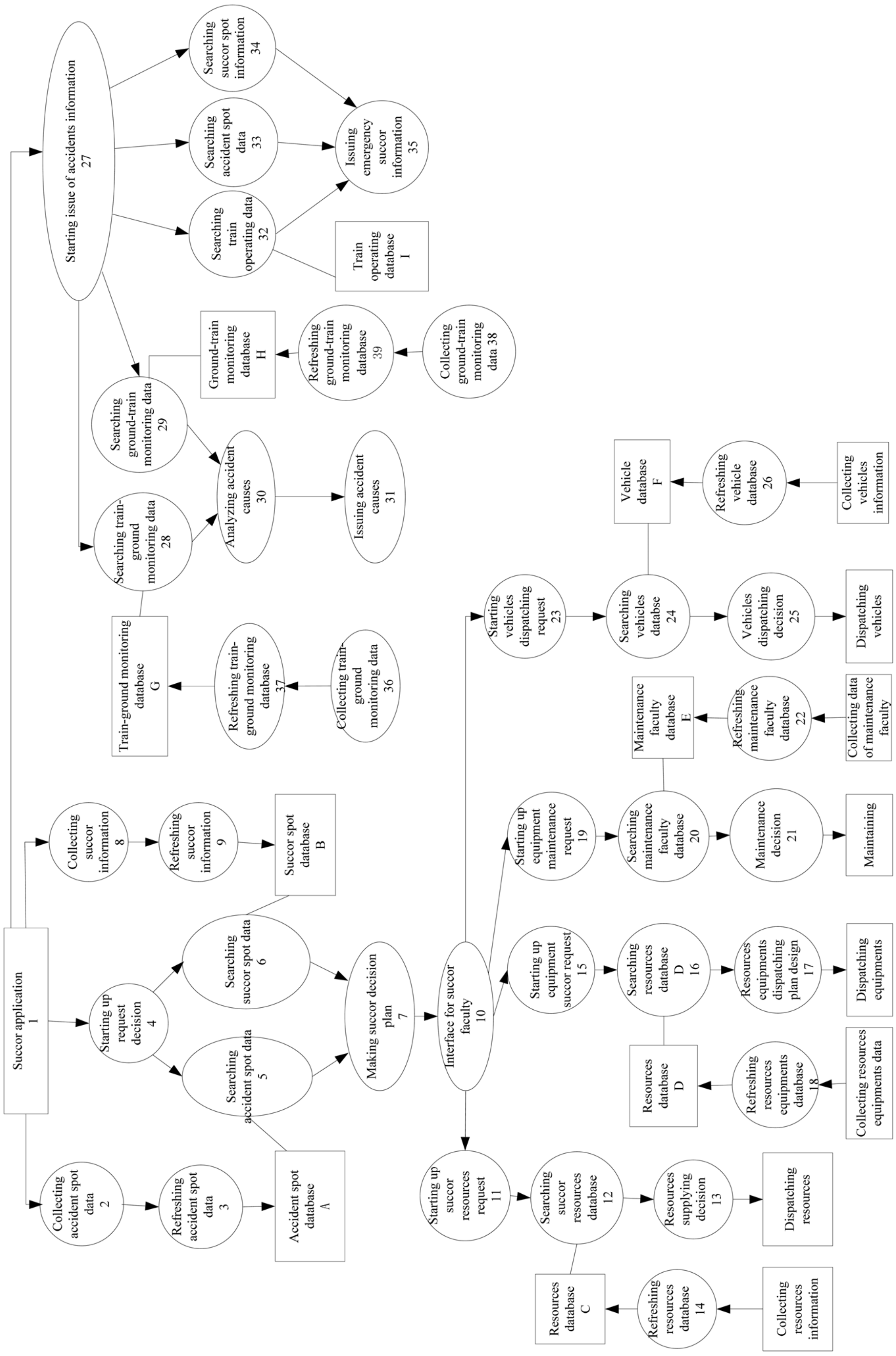

Succor application 1
Collecting accident spot data 2
Refreshing accident spot data 3
Accident spot database A
Starting up request decision 4
Searching accident spot data 5
Searching succor spot data 6
Making succor decision plan 7
Collecting succor information 8
Refreshing succor information 9
Succor spot database B
Interface for succor faculty 10
Starting up succor resources request 11
Searching succor resources database 12
Resources supplying decision 13
Dispatching resources
Resources database C
Refreshing resources database 14
Collecting resources information
Starting up equipment succor request 15
Searching resources database D 16
Resources equipments dispatching plan design 17
Dispatching equipments
Resources database D
Refreshing resources equipments database 18
Collecting resources equipments data
Starting up equipment maintenance request 19
Searching maintenance faculty database 20
Maintenance decision 21
Maintaining
Maintenance faculty database E
Refreshing maintenance faculty database 22
Collecting data of maintenance faculty
Starting vehicles dispatching request 23
Searching vehicles databse 24
Vehicles dispatching decision 25
Dispatching vehicles
Vehicle database F
Refreshing vehicle database 26
Collecting vehicles information
Starting issue of accidents information 27
Searching train-ground monitoring data 28
Searching ground-train monitoring data 29
Analyzing accident causes 30
Issuing accident causes 31
Searching train operating data 32
Searching accident spot data 33
Searching succor spot information 34
Issuing emergency succor information 35
Train operating database I
Train-ground monitoring database G
Refreshing train-ground monitoring database 37
Collecting train-ground monitoring data 36
Ground-train monitoring database H
Refreshing ground-train monitoring database 39
Collecting ground-train monitoring data 38

Fig. 3. DFD of emergency succoring system
Materials supply (meta-service 1):{1,4,(5,6),7,10,11,12,13}

Dispatching of materials and equipments (meta-service 2):{1,4,{5,6},7,10,15,16,17}
Maintaining service (meta-service 3):{1,4,{5,6},7,10,19,20,21}
Dispatching of cars (meta-service 4): {1, 4, {5, 6}, 7, 10, 23, 24, 25}
Issuance of causation of accidents (meta-service 5): {1, 27, ((28, 30), (29, 31))}
Issuance of succoring information (meta-service 6): {1, 27, (32, 33, 34), 35}
Collecting of information of accidents scenes (meta-service 7) :{ 1, 2, 3}
Collecting of succoring information (meta-service 8) :{ 1, 8, 9}
Inspection of trains to ground equipments (meta-service 9): {36, 37}
Inspection of ground equipments to trains (meta-service 10): {38, 39}

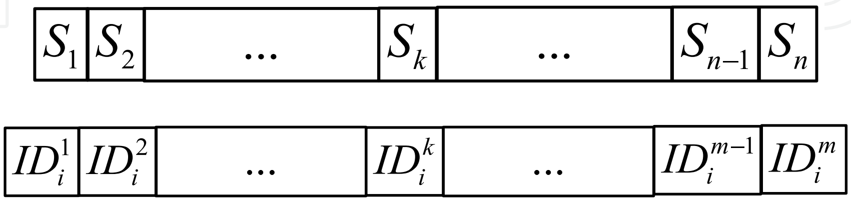


Fig. 4. Coding of Chromosomes for RITS physical structures optimization

(2) Parameters setting

Take it for granted that the all the physical sub-systems have a full-linking relation. That is to say, any sub-system can communicate with all the other sub-systems. There are five kinds of network links today, listed in Table 1. The FDDI (Fiber Distributed Data Interface), Ethernet 1 and Ethernet 2 can be used in LAN (Local Area Network), and DDN (Defense Data Service), T1 link and 56k link can be used in long-distance link.

| Type | Transmission(Mbit/s) |
|---|---|
| FDDI | 103~104 |
| Ethernet 1 | 100 |
| Ethernet 2 | 10 |
| DDN | 2 |
| T1 link | 1.544 |
| 56k link | 56k |

Table 1. Types of the links and the transit speed

To calculate the economic and technical indexes, the other parameters are designed, as shown in Table 2 and Table 3.

| Economic cost | | Computation performance | |
|---|---|---|---|
| hardware cost | communication cost | Parameters about time | Transmission time |
| 3000¥/ Computer | LAN: 0¥/Mbit | Band width:<br>LAN: 100Mbit<br>Internet: 10Mbit | Random delayed time<br>Engorged temporarily |
| | Internet: 8¥/Mbit | Computer performance<br>200MIPS | |

Table 2. Parameters setting in physical structure optimization

| Nodes | Amount of calculation($10^5$MI) | Nodes | Amount of calculation($10^5$MI) |
|---|---|---|---|
| 1 | 1 | 21 | 100 |
| 2 | 30 | 22 | 5 |
| 3 | 5 | 23 | 1 |
| 4 | 3 | 24 | 20 |
| 5 | 20 | 25 | 100 |
| 6 | 20 | 26 | 5 |
| 7 | 100 | 27 | 3 |
| 8 | 30 | 28 | 30 |
| 9 | 5 | 29 | 100 |
| 10 | 10 | 30 | 200 |
| 11 | 1 | 31 | 5 |
| 12 | 20 | 32 | 15 |
| 13 | 100 | 33 | 20 |
| 14 | 5 | 34 | 30 |
| 15 | 1 | 35 | 5 |
| 16 | 20 | 36 | 10 |
| 17 | 100 | 37 | 20 |
| 18 | 5 | 38 | 20 |
| 19 | 1 | 39 | 30 |
| 20 | 20 | 40 | |

Table 3. Amount of data processed of DFD graph nodes

(3) Constraints setting

The constraints are as follows. (1) The number of the equipments constraint. In this case the number is set to be 5. (2) Some of the nodes must be allocated in some certain sub-systems. Thus the optimization process is to allocate other nodes in the sub-systems. The constraints are shown in Table 4.

| Sub-systems( $v_1$ ) | Sub-systems names | Nodes constraints |
|---|---|---|
| $v_1$ | User sub-system | |
| $v_2$ | Central management system | |
| $v_3$ | Train sub-system | 36 |
| $v_4$ | Rail- sideward sub-system | 38 |
| $v_5$ | Station sub-system | |

Table 4. Constraints of nodes positions

(4) Computing results and analysis

The computing results of the example of optimize the physical structure of the emergency succoring system are as follows.

(a) The results of the first calculation

The parameters are as follows. The size of the population *Popsize* =300 and the maximal iteration number is 100. The intersecting probability is 0.9 and the mutation probability is 0.02. The computing results are shown in Figure 5.

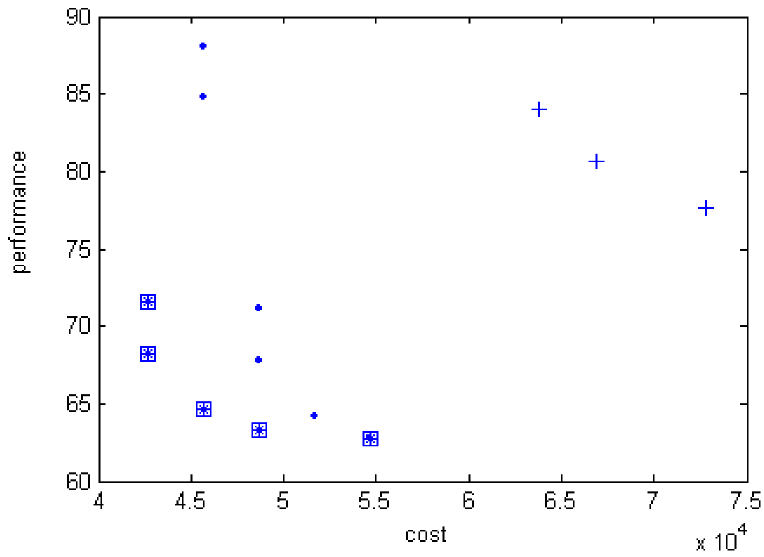


Fig. 5. Results of the first calculation

The problem is a kind of integer programming problem and the non-inferior solutions distribute discretely. '+' denotes the optimal individual chromosome after 10 iterations. '.' denotes the optimal individual chromosome after 60 iterations and '*' for 100. The number of the non-inferior solution is 37.

(b) The results of the second calculation

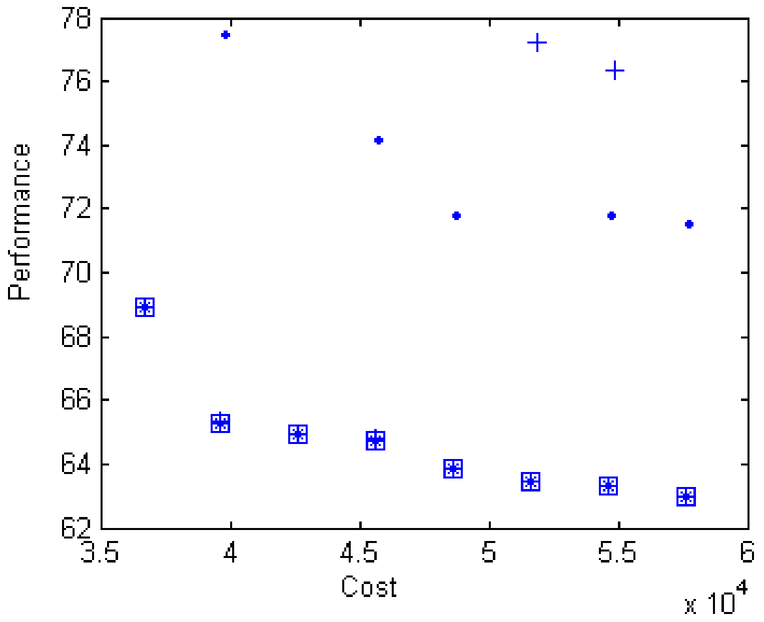


Fig. 6. Results of the second calculation

The parameters are as follows. The size of the population $Popsize = 500$ and the maximal iteration number $Max\_gen$ is 100. The intersecting probability is 0.9 and the mutation probability is 0.02. The computing results are shown in Figure 6.

'+' denotes the optimal individual chromosome after 20 iterations. '.' denotes the optimal individual chromosome after 30 iterations and 'ϒ' for 40, '*' for 100. The number of the non-inferior solution is 53. The typical solution is shown in Table 5.

| | Sub-system 1 | Sub-system 2 | Sub-system 3 | Sub-system 4 | Sub-system 5 | Cost (¥) | Capability (s) |
|---|---|---|---|---|---|---|---|
| 1 | { 1 32 33 35 },<br>{ 27 28 29 34 },<br>{ 30 31}, | { 2 3 },<br>{ 4 5 6 7 },<br>{ 8 9 }, | { 36 37 }, | { 38 39 }, | { 10 11 12 13 15 },<br>{ 16 17 23 },<br>{ 19 20 21 24 25 }, | 45640 | 64.745 |
| 2 | { 1 27 32 33 35 },<br>{ 28 29 34 },<br>{ 30 31}, | { 2 3 },<br>{ 4 5 6 7 },<br>{ 8 9 }, | {36 37 }, | { 38 39 }, | { 10 11 12 13 15 },<br>{ 16 17 23 },<br>{ 19 20 21 24 25 }, | 48640 | 63.845 |
| 3 | { 1 },<br>{ 27 28 29 32 33 34 35 },<br>{ 30 31}, | { 2 3 },<br>{ 4 5 6 7 },<br>{ 8 9 }, | { 36 37 }, | { 38 39 }, | { 10 11 12 13 15 },<br>{ 16 17 23 },<br>{ 19 20 21 24 25 }, | 39640 | 65.29 |
| 4 | { 1 },<br>{ 27 32 33 35 },<br>{ 28 29 },<br>{ 30 31 34 }, | { 2 3 },<br>{ 4 5 6  8 9 },<br>{ 7 }, | { 36 37 }, | { 38 39 }, | { 10 15 16 17 },<br>{ 11 12 13 23 },<br>{ 19 20 21 24 25 }, | 57640 | 62.955 |
| 5 | { 1 }, { 27 28 29 32 33 34 35 },<br>{ 30 31 }, | { 2 3 },<br>{ 4 5 6 7 },<br>{ 8 9 }, | { 36 37 }, | { 38 39 }, | { 10 15 16 17 },<br>{ 11 12 13 23 },<br>{ 19 20 21 24 25 }, | 42640 | 64.915 |
| 6 | { 1 27 32 33 35 },<br>{ 28 29 },<br>{ 30 31 34 }, | { 2 3 },<br>{ 4 5 6 7 },<br>{ 8 9 }, | {36 37 }, | { 38 39 }, | { 10 15 16 17 },<br>{ 11 12 13 23 },<br>{ 19 20 21 24 25 }, | 51640 | 63.47 |
| 7 | { 1 27 32 33 35 },<br>{ 28 29 34 },<br>{ 30 31 }, | { 2 3 },<br>{ 4 5 6  8 9 },<br>{ 7 }, | {36 37 }, | { 38 39 }, | { 10 11 12 13 23 },<br>{ 1516 17 },<br>{ 19 20 21 24 25 }, | 54640 | 63.33 |
| 8 | {1 }, { 27 28 29 32 33 34 35 },<br>{30 31}, | { 2 3 },<br>{4 5 6 7},<br>{ 8 9 }, | {36 37 }, | { 38 39 }, | { 10 11 12 13 },<br>{ 15 16 17 },<br>{ 19 20 21 },<br>{ 23 24 25 }, | 36720 | 68.9 |

Table 5. Optimization mapping results of typical emergency succoring sub-systems

We can see from Table 5 that the distributions of the function units on the system according to different solutions are similar. For example, according to most of the plans, 3 equipments are allocated in sub-system 1, to receive the succoring application, retrieve information from the database and issue the accidents causations and etc. And 3 equipments are allocated in sub-system 2 to search for the succoring information, collect information of accidents scenes and plan for succoring. In addition, the plans for sub-system 3 and for sub-system 4 are the same. An equipment is allocated in the two sub-systems respectively to realize the functions of inspection of trains to ground equipments and inspection of ground equipments to trains. 3 equipments are allocated in sub-system 5 to dispatch the resources, equipments, maintaining services and succoring cars.

To sum up, the solutions are reasonable and the decision makers can choose one of the solutions. A typical solution is shown in Figure 7.

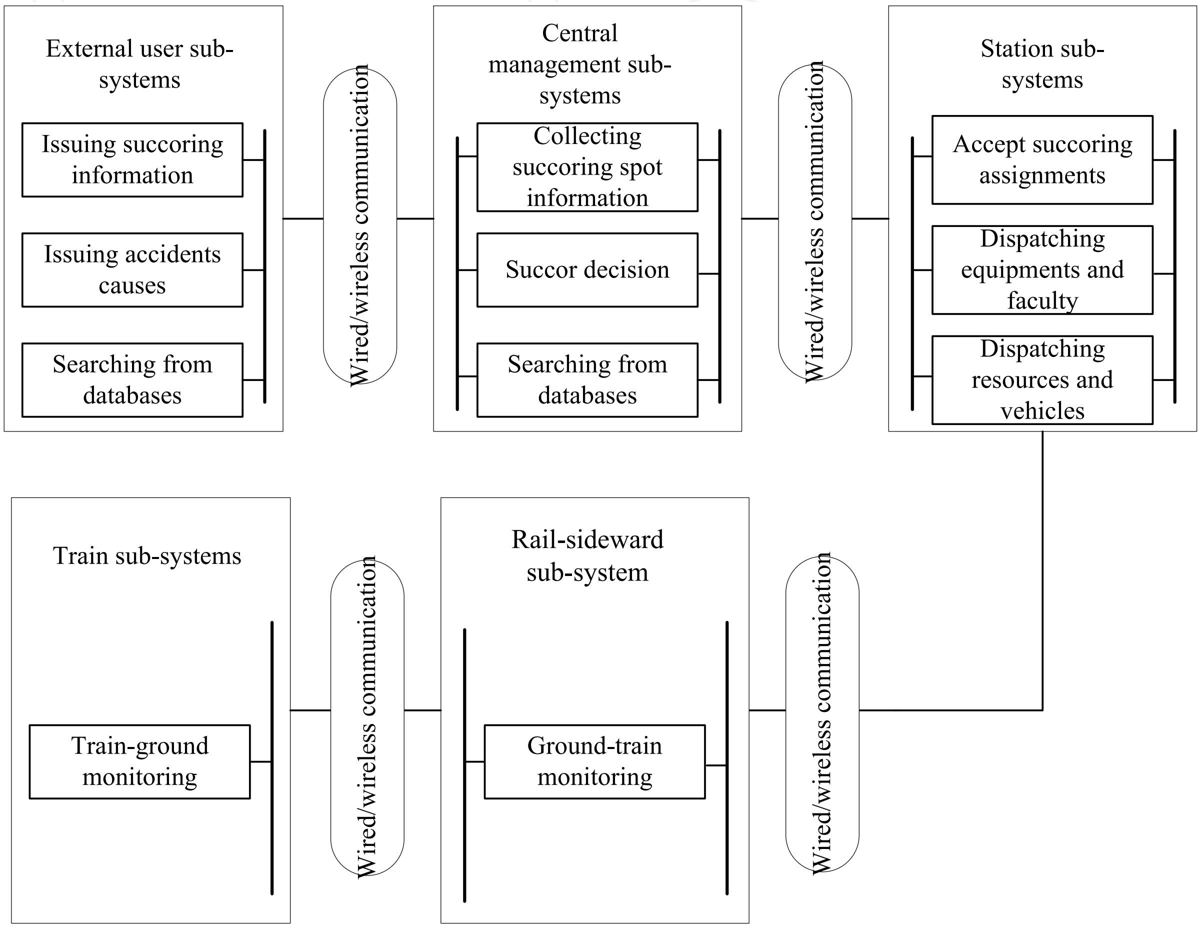

External user sub-systems
Issuing succoring information
Issuing accidents causes
Searching from databases
Wired/wireless communication
Central management sub-systems
Collecting succoring spot information
Succor decision
Searching from databases
Wired/wireless communication
Station sub-systems
Accept succoring assignments
Dispatching equipments and faculty
Dispatching resources and vehicles
Train sub-systems
Train-ground monitoring
Wired/wireless communication
Rail-sideward sub-system
Ground-train monitoring
Wired/wireless communication

Fig. 7. An example of emergency succoring system structure

## 4. Train stops setting optimization

Train stops setting optimization problem is a typical problem of transportation resources allocation. In this case, the railway resources are the trains, which are the most important resources of railway. The coefficient of utilization of this kind of resource determines the service quality of the passenger transportation. So train stops setting is a key problem in railway resources allocation study. In this section, we will introduce GA application in train stops setting problem.

(1) Line planning

Line plan is the allocation of trains respected to passenger transportation demand, which determines the grade, type, original stations and destination stations, number, paths, stops of passenger trains. It is also the basis of the train timetable, which is the most important arrangement of resources for railway transportation organization.

General line planning is an extremely complex and difficult problem; even its sub-problems considered separately are already NP-hard from a computational complexity point of view. Its process is based on the following input: a public demand, the available railway line, a set of trains. The goal is to obtain a set of lines and associated timetables to which trains are assigned.

A typical sub-problem of line planning is to generate the stops plan of trains. A most valuable advantage of GA in line planning is that the codes in chromosome can exactly describe whether the trains will stop at all the stations, 0 standing for non-stop and 1 for stop. And the stop operation will cause costs, while it meets the passenger transportation demand. The costs and the benefits caused by meeting the passenger transportation demand can both be formulated by the chromosomes codes, generating the adaptive function formulas. Thus, the optimizing process is to find minimal costs adaptive function value and maximal benefits adaptive function value through the iterations.

### 4.1 Designing the stops of the trains

The problem to select the stops of a train is very hard. It can be described as a planning problem, with 0 standing for the train does not stop at a station and 1 denoting the train stops at a station. The searching space is very large and the decision variables are in great volume. So the heuristic algorithm has the superiority to solve this problem. Genetic algorithm is hired (Zha, Liu and Li 2008) and Simulate Anneal Arithmetic algorithm is used to solve the problem (Wang, Peng and Xie 2006). Tabu Searching algorithm is hired (Deng, Shi and Zhou 2009).

The steps of the algorithm are follows.

**Step 1.** To design the code of the problem;
**Step 2.** To initialize the population $X(0)=(x_1,x_2,.....x_n)$;
**Step 3.** To calculate the value of adaptive function $F(x_i)$ of each chromosome in population $X(t)$;
**Step 4.** To create the middle generation $X_r(t)$;
**Step 5.** To create the new generation $X(t+1)$ based on the middle generation $X_r(t+1)$;
**Step 6.** $t=t+1$; If the exiting condition does not exist, go step 4.

The algorithm is designed for the train stops setting problem as follows.

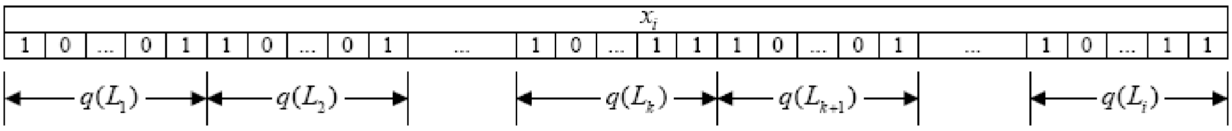
(1) Solution coding of the problem

Coding is to express the feasible solution as a characteristic string, which can describe the characteristic of the problem. And it is required that the codes are easy to be dealt with. The train stops can be coded as string and there is a 0 or 1 at each bit. 0 denotes that the train will not stop at a certain station and 1 denotes that it will. When the strings of all the trains are linked, a chromosome is formed.

In this section, the code of a single train stops is designed as a one-dimensional array $x_i$. The length is *i*.

$$i = \sum_{j=1}^{p} q\left(L_j\right) \tag{7}$$

$p$ is the number of the sections that the trains run through. $q\left(L_j\right)$ denotes the number of the stops when train $j$ runs through section $L_j$. The code of a chromosome is constructed as shown in Figure 8.



| $x_i$ | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | ... | 0 | 1 | 1 | 0 | ... | 0 | 1 | ... | 1 | 0 | ... | 1 | 1 | 1 | 0 | ... | 0 | 1 | ... | 1 | 0 | ... | 1 | 1 | |

Fig. 8. Coding representation

(2) Initialization of the population

Initialization of the population is to construct the original population as the initial solution of the problem. To create the population of the initial solution is to generate *pop_size* chromosome, which is shown in Figure 8. And it is a constraint to meet the station capacity requirement. So it is necessary to judge whether the capacity constraint is satisfied during generating of the initial solution.

(3) Evaluating of the population

It is a must to evaluate the population to judge the quality of the population. The index is the value of the adaptive function. The adaptive function value is the symbol of the adaptability of the chromosome. The bigger the function value is, the more opportunity to survive the chromosome has. The adaptive function and the objective function are closely related to each other. The objective function can be seen as the mutation of the adaptive function. Generally speaking, the adaptive function can be used as the objective function when the objective function is non-negative or it is a maximal optimizing problem. When the objective function is negative or it is a minimum optimizing problem, the objective function can be designed as an equation that an enough large real number minus the objective functions. In this section, the passenger distributing speed is taken as the objective function and it is non-negative. And it is a maximal optimizing problem, so the adaptive function is designed to be equal to the objective function.

(4) Selection

Selection is to select an outstanding chromosome from a generation and past its excellent gene to the next generation. The roulette method is often taken in programming. Roulette method is a proportion strategy. Its main idea is to select the chromosome by the adaptive function value. The four steps of roulette are as follows.

a. To calculate the adaptive value $eval\left(v_k\right)$ of every chromosome $v_k$.

b. To calculate the summary of the adaptive value of all the chromosomes.

c. To calculate the selecting probability $p_k$ of chromosome $v_k$.

d. To calculate the accumulated probability $q_k$ of every chromosome $v_k$.

Then a plate is formed and it is cycled for $pop\_size$ times. Every time one chromosome is selected to create the new generation.

a. To generate a pseudo-random number $r \in [0,1]$.

b. If $r \le q_1$, then select the first chromosome $v_1$; Otherwise, select the $k$th chromosome $v_k$ to meet the requirement $q_{k-1} \le r \le q_k$.

(5) Crossover.

The crossover operation includes one-point crossover, multi-point crossover, part- crossover and etc. In this section the one-point crossover is hired. To select a single one-point place for the two father chromosomes, then from the crossover place, the two chromosome exchange the value of every bit till the end of the chromosome.

(6) Mutation.

The task of the mutating is to change the value of a certain bit. In this section, mutating is to negate on the bit. That is to say, to change 1 into o and 0 to 1. It has two steps. The first step is to select a place to determine the bit to execute mutation operation. The second is to execute the negating operation. Then the new generation is created after step (2) to step (6).

(7) To judge if the loop can be cancelled. If it can then exit, else go step (2).

### 4.2 Calculating the number of the two types of the trains

In this section, a multiple unit contains 16 cars and its capacity $A_c$ is 1200. The low speed train contains 17 cars and has the capacity of 1400. We first calculate out the numbers of the two types of the trains, which is shown in Table 6.

| Running section | Number of the two types of trains | |
|---|---|---|
| | Multi Unit | Low speed trains |
| Shangqiu--Xuzhou--Bengbu—Nanjing | 0 | 3 |
| Shangqiu--Xuzhou—Bengbu | 0 | 3 |
| Xuzhou--Bengbu—Nanjing | 52 | 4 |
| Bengbu--Huainan--Hefei | 0 | 3 |
| Fuyang--Huainan--Hefei | 0 | 3 |
| Hefei--Nanjing | 2 | 1 |
| Xuzhou—Shangqiu | 0 | 6 |
| Xuzhou—Bengbu | 9 | 0 |
| Xuzhou--Xinyi | 0 | 3 |
| Bengbu—Nanjing | 0 | 5 |
| Fuyang—Huainan | 0 | 4 |
| Huainan--Hefei | 0 | 3 |
| Xuzhou--Shangqiu--Fuyang--Huainan--Hefei | 0 | 2 |
| Shangqiu--Xuzhou--Xinyi | 0 | 2 |
| Shangqiu--Fuyang--Huainan--Hefei | 0 | 4 |
| Yangzhou—Nanjing | 0 | 1 |
| Fuyang--Huainan--Hefei—Nanjing | 0 | 1 |

Table 6. Running sections, numbers and grades for train operation under accident disaster

### 4.3 Stops setting

We first set the intersecting probability to be 0.6 and mutation probability to be 0.1. The maximal iteration number is 10000.

The adaptive function raised value to its peal value 263.1k after 580 iterations. Then we get the stops plan through interpreting the best chromosome. The information about the stops is shown in Table 7.

| Running section | Number of the two kind of trains | | Stop plan |
|---|---|---|---|
| | G | T | |
| Xuzhou--Shangqiu--Fuyang--Huainan--Hefei--Nanjing | 0 | 22 | One stops at Fuyang,,<br>one stops at Hefei,<br>others have no stop. |
| Xuzhou--Xinyi--Yangzhou—Nanjing | 0 | 2 | One stops at Yangzhou,<br>another has no stop. |
| Xuzhou--Bengbu—Nanjing | 28 | 8 | 5 stop at Bengbu |
| Shangqiu--Xuzhou--Bengbu—Nanjing | 0 | 3 | 3 stop at Xuzhou. |
| Shangqiu--Xuzhou—Bengbu | 0 | 3 | All have no stops. |
| Bengbu--Huainan—Hefei | 0 | 3 | All have no stops. |
| Fuyang--Huainan—Hefei | 0 | 9 | 7 stop at Huainan. |
| Hefei--Nanjing(high speed rail) | 2 | 0 | All have no stops. |
| Xuzhou—Shangqiu | 0 | 6 | All have no stops. |
| Xuzhou--Bengbu(high speed rail) | 9 | 0 | All have no stops. |
| Xuzhou—Xinyi | 0 | 3 | All have no stops. |
| Xuzhou--Shangqiu--Fuyang--Huainan—Hefei | 0 | 6 | 4 stop at Shangqiu. |
| Shangqiu--Xuzhou—Xinyi | 0 | 2 | All have no stops. |

Table 7. Train operation plan for section Xuzhou-Bengbu

## 5. Train rescheduling

The dynamic transportation resources allocation optimization is to give online, real time solution of the resources allocation problem to assure the railway transportation system operate safely. Train rescheduling is a typical dynamic transportation resources optimization problem.

Train rescheduling is to reset the inbound and outbound time of trains at stations when the operating plan is affected by disturbances. Similar with timetable optimization, the interval time and buffer time must be considered to assure the feasibility and stability of the timetable. And most of the constraints in train rescheduling are as the same as that in timetable optimization problem. The difference is that there is disturbance, and the train rescheduling is an online dynamic process, which requires more efficient algorithm to give solution in time to dispatch the trains.

A typical application of GA in dynamical transportation resources re-allocation is train rescheduling. The approach to apply GA in train rescheduling is very similar with that in timetable optimization. We will hire the variation of GA to obtain higher calculating efficiency, generating satisfactory solution, not necessarily the optimal solution.

### 5.1 The mathematic model of train rescheduling problem

The variables are as follows.

$m$: number of stations in the dispatching section.

$n$: number of trains that needs to be rescheduled.

$g$*: the planned timetable. $g* = \left\{\left(g_{i,j}^{a,*}, g_{i,j}^{d,*}\right) | i = 1,2,.....,n; j = 1,2,....m\right\}, g_{i,j}^{a,*}$, is the arriving time of train $i$ at station $j$ according to $g*$. $g_{i,j}^{d,*}$ is the departing time of train $i$ from station $j$ according to $g*$.

G: The set that is made of all the timetables. $G = \{g \mid g \text{is a possible timetable}\}$,

$g = \left\{\left(g_{i,j}^{a}, g_{i,j}^{d}\right) | i = 1,2,.....,n; j = 1,2,....m\right\}, g_{i,j}^{a}$, is the arriving time of train $i$ from station $j$ according to $g$ . $g_{i,j}^{d}$ is the departing time of train $i$ at station $j$ according to $g$ .

Set $\forall p,q \in G$ .Then the *distance* between $p$ and $q$ is

$$d(p,q) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sqrt{\left(p_{i,j}^{a} - q_{i,j}^{a,*}\right) + \left(p_{i,j}^{d} - q_{i,j}^{d,*}\right)} \tag{8}$$

Tthe destination of train rescheduling problem is to minimize $d(p,q*)$ .

It can be described as follows.

$$f = _{\forall p \in G}^{\min d\left(p,g^{*}\right)} \tag{9}$$

That is to say, the goal of train rescheduling is to find a timetable which has the minimal distance from the planned timetable. And the trains are clarified into different grades, the goal is changed, added a weight variable $w_i$ . The goal is described as follows.

$$\min f = \sum_{i=1}^{n}\sum_{j=1}^{m} w_i \bullet \sqrt{\left(p_{i,j}^{a} - g_{i,j}^{a,*}\right) + \left(p_{i,j}^{d} - g_{i,j}^{d,*}\right)} \tag{10}$$

Note: $w_1 > 0$ . The bigger $w_1$ is, the higher grade train $i$ belongs to. $p_{i,j}^{a}$ and $p_{i,j}^{d}$ are the decision variables. Now we discuss the constraints.

First, the departing time can not be earlier than that in the planned timetable. So

$$p_{i,j}^{d} \geq g_{i,j}^{d,*}\left(j = 1,2,...,m\right) \tag{11}$$

$$\text{Set } s_{i,j}^{t} = \begin{cases} 1, \text{train } i \text{ occupyingarail line of thestation} \\ 0, \text{otherwise} \end{cases}$$

If there are $D_j$ rail lines in the station, then

$$\sum_{i=1}^{n} s_{i,j}^{t} \le D_j \tag{12}$$

Another constraint is about the running time of the trains in the sections.

$$p_{i,j+1}^{a} \ge p_{i,j}^{d} + T_{\min}\left(j = 1,2,...,m-1\right) \tag{13}$$

$T_{\min}$ is the minimal running time of train $i$ in section $[j,j+1]$.

According to the minimal headway time constraint,

$$| p_{i,j}^{d} - p_{k,j}^{d} | \ge I\left(i \ne k, j = 1,2,....,m\right) \tag{14}$$

$I$ is the minimal headway time.

And there must be skylight time reserved to do the maintaining work. So

$$p_{i,j}^{a}, p_{i,j}^{d} \le t_b \ \text{ or } \ p_{i,j}^{a} \ge t_c\left(i = 1,2,...,n; j = 1,2,...,m\right) \tag{15}$$

$t_b$ is the beginning time of skylight time and $t_c$ is the end time of skylight time.

### 5.2 Train rescheduling problem solving based on genetic algorithm

The train rescheduling problem is actually nonlinear optimization problem with constraints. The problem can be solved with genetic algorithm (H. Wang, Q. Zhang, J. Wang, Z. Wang and Y. Zhang 2006). Now we design the algorithm for it.

### 5.2.1 Coding design

The train rescheduling problem is to reset the arriving and departing time of trains to optimize the goal. To simplify the description of the time, number of minutes from zero o'clock are designed to represent the arriving time and departing time. For example, 01:06 can be replaced with 66, for 66 is the number of minutes from 00:00 to 01:06. So the chromosome can be coded with such integers. The chromosome is shown in Figure 9. And it can be seen that the maximal length is $2nm$ .

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $p_{1,1}^{a}$ | $p_{1,1}^{d}$ | … | $p_{1,m}^{a}$ | $p_{1,m}^{d}$ | … | $p_{i,m}^{a}$ | $p_{i,m}^{d}$ | … | $p_{n,m}^{a}$ | $p_{n,m}^{d}$ |

Fig. 9. Coding of chromosome in train rescheduling problem

### 5.2.2 The fitness function

The fitness function is a function that evaluates the ability that the chromosome is adaptive to the environment. The bigger the fitness function value, the greater ability the chromosome has.

According to constraint described in equation (10), the departing time can not be earlier that the planned departing time. So this constraint can be dealt with penalty function.

Constraints described in equation (11) can be adjusted to be adapted with the genetic algorithm.

Function $f_1$ is defined as follows.

$$f_1 = \begin{cases} 0, \sum_{i=1}^{n} s_{i,j}^{t} \le D_j \left(j=1,2,\ldots,m\right) \\ M, \text{otherwise} \end{cases} \tag{16}$$

Function $f_2$ is defined for constraint described in equation (12) as follows.

$$f_2 = \begin{cases} 0, p_{i,j+1}^{a} \ge p_{i,j}^{d} + T_{\min} \left(i=1,2,\ldots,m\right) \\ M, \text{otherwise} \end{cases} \tag{17}$$

Function $f_3$ is defined for constraint described in equation (13) as follows.

$$f_3 = \begin{cases} 0, |p_{i,j}^{d} - p_{k,j}^{d}| \ge I\left(i \ne k, j=1,2,\ldots,m\right) \\ M, \text{otherwise} \end{cases} \tag{18}$$

Function $f_4$ is defined for constraint described in equation (14) as follows.

$$f_4 = \begin{cases} 0, p_{i,j}^{a}, p_{i,j}^{d} \le t_b \text{ or } \mathrm{p}_{i,j}^{a} \ge t_c \left(i=1,2,\ldots,n.j=1,2,\ldots,m\right) \\ M, \text{otherwise} \end{cases} \tag{19}$$

$M$ is a positive integer.

Then the fitness function can be defined as follows.

$$\begin{aligned} f\left(\vec{p}_1, \vec{p}_2, \ldots, \vec{p}_i, \ldots \vec{p}_n\right) = C_{\max} - \sum_{i=1}^{n} \sum_{j=1}^{m} w_i \bullet \sqrt{\left(p_{i,j}^{a} - g_{i,j}^{a,*}\right)^2 + \left(p_{i,j}^{d} - g_{i,j}^{d,*}\right)^2} \\ - \sum_{i=1}^{n} \sum_{j=1}^{m} w \bullet \sqrt{\left(p_{i,j}^{d} - g_{i,j}^{d,*}\right)^2} \operatorname{sgn}\left(p_{i,j}^{d} - g_{i,j}^{d,*}\right) - f_1 - f_2 - f_3 - f_4 \end{aligned} \tag{20}$$

$C_{\max}$ is the maximal positive integer that the computers can generate. $\vec{p}_i = \left(p_{i,1}^{a}, p_{i,1}^{d}, \ldots, p_{i,m}^{a}, p_{i,m}^{d}\right), i = 1,2,\ldots,n$; $w$ is the penalty factor of constraint described in equation (11). S gn is the sign function.

In fact, constraints described in equation (12) to (15) are also processed with penalty function where $M$ is the penalty factor.

### 5.2.3 Crossover operator

The crossover operation is the main characteristic that genetic algorithms are different from other algorithms, which is the method to generate the next generation of chromosomes. The two-point intersecting approach is taken here, as shown in Figure 10.

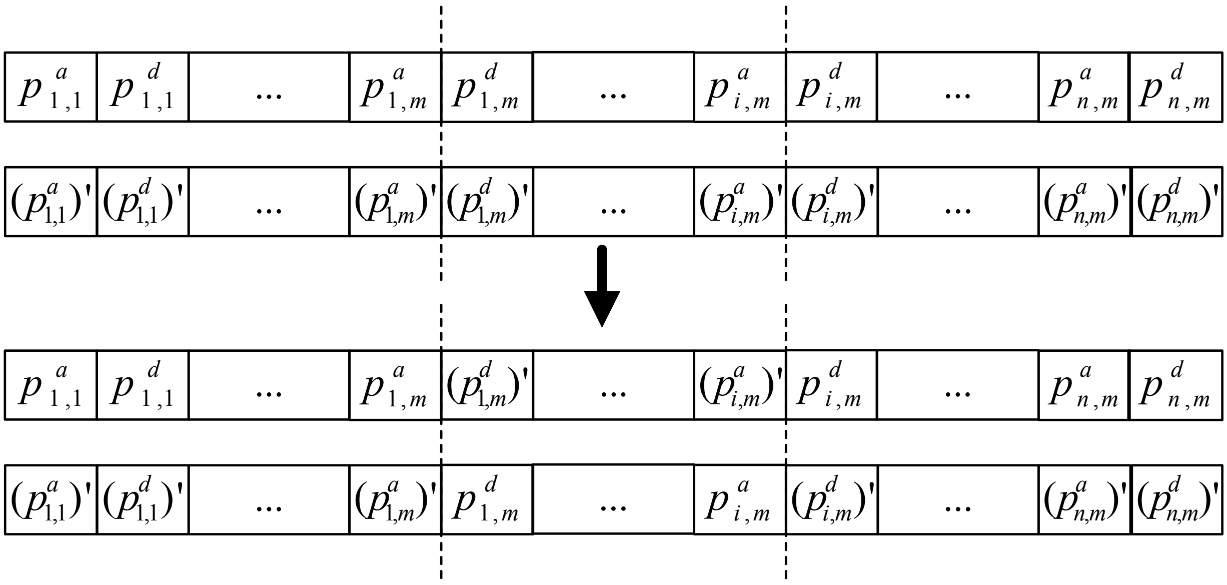

$p^a_{1,1}$ $p^d_{1,1}$ ⋯ $p^a_{1,m}$ $p^d_{1,m}$ ⋯ $p^a_{i,m}$ $p^d_{i,m}$ ⋯ $p^a_{n,m}$ $p^d_{n,m}$
$(p^a_{1,1})'$ $(p^d_{1,1})'$ ⋯ $(p^a_{1,m})'$ $(p^d_{1,m})'$ ⋯ $(p^a_{i,m})'$ $(p^d_{i,m})'$ ⋯ $(p^a_{n,m})'$ $(p^d_{n,m})'$
$p^a_{1,1}$ $p^d_{1,1}$ ⋯ $p^a_{1,m}$ $(p^d_{1,m})'$ ⋯ $(p^a_{i,m})'$ $p^d_{i,m}$ ⋯ $p^a_{n,m}$ $p^d_{n,m}$
$(p^a_{1,1})'$ $(p^d_{1,1})'$ ⋯ $(p^a_{1,m})'$ $p^d_{1,m}$ ⋯ $p^a_{i,m}$ $(p^d_{i,m})'$ ⋯ $(p^a_{n,m})'$ $(p^d_{n,m})'$

Fig. 10. Crossover pattern

### 5.2.4 Mutation operator

Mutation is to avoid the premature of the chromosomes, which makes some of the genes in chromosomes to mutate according to a certain probability. Here the mutation is designed as follows.

$$S: p \to p + \alpha, \alpha \in [-30, 30] \tag{21}$$

$\alpha$ is a variable which distributes uniformly. That is to say, the mutation operator disturbs $p$ with a certain probability and the range is less than 1 minute.

### 5.3 Case study

To prove the feasibility of the algorithm, we propose a case and analyze the computing results. The basic data is taken from the real train timetable of China railway, which makes the case more credible and proves the feasibility of algorithm more directly.

### 5.3.1 Basic parameters

(1)Number of stations M=6,(2)Number of trains N=11,(3)Train level,(4) The minimal running time of different types of trains,(5) The added time for train stopping operation,(6) The added time for train speeding operation,(7) The operating time of four types of trains (min),(8) The interval time $I$ =4min,(9) The skylight: 0-6 h,(10) The weights of four types of trains $w = [4\ 3\ 2\ 1]$.

$$R_{\min} = \begin{bmatrix} 20 & 24 & 30 & 36 \\ 25 & 32 & 39 & 48 \\ 29 & 36 & 44 & 54 \\ 9 & 11 & 14 & 17 \\ 23 & 28 & 35 & 44 \end{bmatrix} \quad T = \begin{bmatrix} 2 & 2 & 2 & 3 \\ 0 & 0 & 0 & 0 \\ 3 & 3 & 3 & 4 \\ 0 & 0 & 0 & 3 \\ 2 & 2 & 2 & 4 \end{bmatrix}$$

The planned timetable is shown in Table 8.

| | | Trains | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Stations | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 1 | Arrive | 7:00 | 7:07 | 7:14 | 7:21 | 8:00 | 8:10 | 8:20 | 8:30 | 9:00 | 9:10 | 9:20 |
| | Depart | 7:06 | 7:13 | 7:20 | 7:27 | 8:08 | 8:18 | 8:28 | 8:38 | 9:06 | 9:16 | 9:26 |
| 2 | Arrive | 7:27 | 7:34 | 7:41 | 7:48 | 8:46 | 8:56 | 9:06 | 9:16 | 9:33 | 9:43 | 9:53 |
| | Depart | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 3 | Arrive | 7:53 | 8:00 | 8:07 | 8:14 | 9:38 | 9:48 | 9:58 | 10:07 | 10:11 | 10:17 | 10:27 |
| | Depart | -- | -- | -- | -- | -- | -- | -- | 10:31 | -- | -- | -- |
| 4 | Arrive | 8:23 | 8:30 | 8:37 | 8:44 | 10:34 | 10:44 | 10:54 | 11:26 | 10:58 | 11:02 | 11:08 |
| | Depart | 8:29 | 8:36 | 8:43 | 8:50 | 10:42 | 10:52 | 11:18 | 11:34 | 11:04 | 11:08 | 11:14 |
| 5 | Arrive | 8:40 | 8:47 | 8:54 | 9:01 | 11:00 | 11:10 | 11:36 | 11:52 | 11:24 | 11:28 | 11:32 |
| | Depart | -- | -- | -- | -- | 11:08 | 11:18 | 11:44 | 12:00 | -- | -- | -- |
| 6 | Arrive | 9:06 | 9:13 | 9:20 | 9:27 | 11:54 | 12:03 | 12:30 | 12:46 | 12:07 | 12:11 | 12:15 |
| | Depart | 9:12 | 9:19 | 9:26 | 9:33 | 12:02 | 12:05 | 12:38 | 12:54 | 12:13 | 12:17 | 12:21 |

Notes:

"—" denotes that the departing time of a train at a certain station is the same with the arriving time.

Then, it is assumed that the fifth train arrived at the first station five minutes later than it is planned in the timetable and other trains run according to the planned timetable. Then we began to reschedule the timetable. First, the rescheduling rules are listed as follows.

Table 8. The planned timetable

### 5.3.2 Rescheduling rules

There are many rules in train rescheduling, which should be applied in the algorithm. Here are the main rules.

1. The grades of the trains can not be changed in rescheduling
2. The higher grades trains can surpass the lower grades trains and the trains of the same grade can not surpass each other
3. A higher grade train can surpass many lower grade trains and a lower grade train can be surpassed by many higher grades trains.

### 5.3.3 Parameters setting

The population size of the chromosomes is 30, the crossover probability is 0.8 and the mutation probability is 0.001. The maximal number of iteration is 100.

### 5.3.4 Results and analysis

The planned timetable is turned to the style described in section 4.2.1. And the timetable is rescheduled according to the method proposed in this section. Then, the arriving time and departing time are turned into the original time style, as shown in Table 9.

| | | Trains | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Stations | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 1 | Arrive | 7:00 | 7:07 | 7:14 | 7:21 | 8:05 | 8:10 | 8:20 | 8:30 | 9:00 | 9:10 | 9:20 |
| | Depart | 7:06 | 7:13 | 7:20 | 7:27 | 8:11 | 8:18 | 8:28 | 8:38 | 9:06 | 9:16 | 9:26 |
| 2 | Arrive | 7:27 | 7:34 | 7:41 | 7:48 | 8:47 | 8:56 | 9:06 | 9:16 | 9:33 | 9:43 | 9:53 |
| | Depart | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- | -- |
| 3 | Arrive | 7:53 | 8:00 | 8:07 | 8:14 | 9:38 | 9:48 | 9:58 | 10:07 | 10:11 | 10:17 | 10:27 |
| | Depart | -- | -- | -- | -- | -- | -- | -- | 10:31 | -- | -- | -- |
| 4 | Arrive | 8:23 | 8:30 | 8:37 | 8:44 | 10:34 | 10:44 | 10:54 | 11:26 | 10:58 | 11:02 | 11:08 |
| | Depart | 8:29 | 8:36 | 8:43 | 8:50 | 10:42 | 10:52 | 11:18 | 11:34 | 11:04 | 11:08 | 11:14 |
| 5 | Arrive | 8:40 | 8:47 | 8:54 | 9:01 | 11:00 | 11:10 | 11:36 | 11:52 | 11:24 | 11:28 | 11:32 |
| | Depart | -- | -- | -- | -- | 11:08 | 11:18 | 11:44 | 12:00 | -- | -- | -- |
| 6 | Arrive | 9:06 | 9:13 | 9:20 | 9:27 | 11:54 | 12:03 | 12:30 | 12:46 | 12:07 | 12:11 | 12:15 |
| | Depart | 9:12 | 9:19 | 9:26 | 9:33 | 12:02 | 12:25 | 12:38 | 12:54 | 12:13 | 12:17 | 12:21 |

The adaptive function value is 10.

Table 9. Computing result of timetable in original time style

We can see from Table 9 that the operating time at station is compressed to 6 minutes, while the planned operating time is 8 minutes, which requires the station workforce to fasten the operation work. And it is still 3 minutes late when it departs from the first station. Then the running time from the first station to the second station is compressed to 36 minutes and the train is re-planned to arrive at the second station at 8:11. Thus it is only 1 minute late. The operating time is not changed at the second station. The train recovers to operate according to the planned timetable at the third station.

It can be concluded that the rescheduled timetable is reasonable and executable from the analysis above. And the algorithm is feasible and practical and the method can be embedded in the train dispatching system to support the operation work.

## 6. Conclusion of this chapter

The main findings are as follows.

1. The improvement to GA brings high calculating precision and efficiency, meeting the calculating requirements of large scale optimization problem with combinatorial complexity and nonlinearity.
2. GA is suitable for railway application oriented problems solving. Although the solution may not be the optimal solution, the calculating time cost is much more acceptable.
3. GA is very easy to apply in optimization problem, not only reflecting in decision variables description, but also in objective function formulas.

And GA can also be applied in many other fields of railway applications, such as the crew scheduling, station operating plan optimization etc. Its optimization ability affords the possibility to solve the optimization problems in railway applications with high precision and efficiency. It is no doubt that GA has a bright future in the field of railway applications.

## 7. References

Honggang Wang, Qin Zhang, Jianying Wang, Zhuangfeng Wang, Yijun Zhang. "GA-based model of train operation adjustment for high-speed railway. China railway science." *China railway science* 27: 3(May 2006): 96-100.

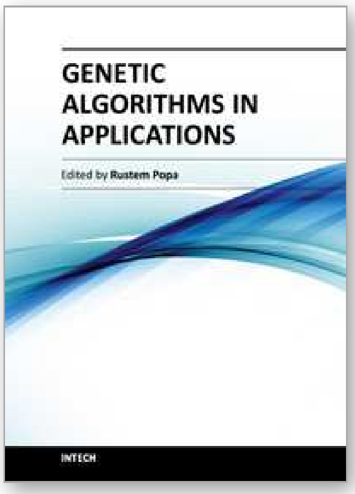
Lianbo Deng, Feng Shi, Wenliang Zhou. "Stop Schedule Plan Optimization for Passenger Train." *China Railway Science* 30:4(July 2009): 102-109.

Thomas Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford County: Oxford University Press,1996.

WeiXiong Zha, Huilin Liu, Jian Li. *Theory and Example of Railway Passengers and Trains Flow*. Chengdu: Southwest Jiaotong University Press, 2008.

Yan Meng. *Structure design methodology for the railway intelligent transportation systems*. Beijing: China Academy of railway sciences, 2005.

Zhi Wang, Qiyuan Peng, Xiaosong Xie. "Research on Optimization of Railway Passenger Train Path Based on Genetic Algorithms." *Railway Computer Application* 15:12 (December 2006): 4-6.



**Genetic Algorithms in Applications**

Edited by Dr. Rustem Popa

ISBN 978-953-51-0400-1
Hard cover, 328 pages
**Publisher** InTech
**Published online** 21, March, 2012
**Published in print edition** March, 2012

Genetic Algorithms (GAs) are one of several techniques in the family of Evolutionary Algorithms - algorithms that search for solutions to optimization problems by "evolving" better and better solutions. Genetic Algorithms have been applied in science, engineering, business and social sciences. This book consists of 16 chapters organized into five sections. The first section deals with some applications in automatic control, the second section contains several applications in scheduling of resources, and the third section introduces some applications in electrical and electronics engineering. The next section illustrates some examples of character recognition and multi-criteria classification, and the last one deals with trading systems. These evolutionary techniques may be useful to engineers and scientists in various fields of specialization, who need some optimization techniques in their work and who may be using Genetic Algorithms in their applications for the first time. These applications may be useful to many other people who are getting familiar with the subject of Genetic Algorithms.

Jia Li-Min and Meng Xue-Lei (2012). Genetic Algorithms Implement in Railway Management Information System, Genetic Algorithms in Applications, Dr. Rustem Popa (Ed.), ISBN: 978-953-51-0400-1, InTech, Available from: http://www.intechopen.com/books/genetic-algorithms-in-applications/genetic-algorithms-implement-in-railway-management-information-system

INTECH
open science | open minds

**InTech Europe**
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

**InTech China**
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821