Blocking time reduction for level crossings using the genetic algorithm

Y. Noguchi[1], H. Mochizuki[1], S. Takahashi[1], H. Nakamura[1], S. Kaneko[1] & M. Sakai[2]
[1]*Nihon University, Japan*
[2]*The Nippon Signal Co., Ltd., Japan*

Abstract

The blocking time of a level crossing influences traffic on a road that crosses a rail line. The blocking time of a level crossing will be especially long on a heavy traffic rail line. The purpose of this paper is to reduce the blocking time of level crossings by optimising the railroad schedule. We propose an optimal schedule in which the departure time at each station is delayed minutely from the time of the planned schedule. Since there are many trains on a heavy traffic rail line, the number of combinations of minutely adjusted departure times will be enormous. We used the genetic algorithm (GA) for searching for an optimal railroad schedule. The delay time for each train at each station is used as a gene value, and a chromosome is composed by connecting the genes for each train. The fitness value is the total blocking time of all level crossings on the model rail line. Results for the computer simulation show that our optimal railroad schedule gives a shorter total blocking time compared with the planned schedule. In addition, we report our examination of a composition of the GA calculation with a grid computing technology for the purpose of reducing the GA operation time
Keywords: level crossing, railroad schedule, genetic algorithm, grid computing.

1 Introduction

At present, on Japanese railroads, an increase of passenger transportation capacity is brought about by an increase in the number of trains and an increase in the number of cars to a train. This leads to lengthening the time that a level crossing is blocked by trains going up and down. As a result, there is a problem


WIT Transactions on The Built Environment, Vol 88, © 2006 WIT Press
www.witpress.com, ISSN 1743-3509 (on-line)
doi:10.2495/CR060301

that a level crossing is often kept blocked for a long time. There are some reported measures to meet this problem, such as an overhead crossing, but the cost of those measures is high [1].

In this paper we examine a way to construct a railroad schedule in view of reducing the blocking time of level crossings. Specifically, we calculated an optimal schedule that minimizes blocking time by varying the time duration for a train to be standing at a station. And because the number of combinations of the time duration is enormous, in the calculation of an optimal schedule we applied the genetic algorithm (GA) that is used for optimization of the problem of combinations.

In addition, in order to reduce the GA operation time, we examined the composition of the calculation system with a grid computing technology that is attracting attention as one of the distributed processing methods.

2 Optimization method of a railroad schedule in view of reducing blocking time of level crossings

2.1 Definition of blocking time of a level crossing

A level crossing is controlled by two controllers, one of which is installed at the point 1 km before the level crossing and the other at the point just after the level crossing, as shown in Figure 1. And the time that a train is running between the two controllers is defined as the blocking time of a level crossing.

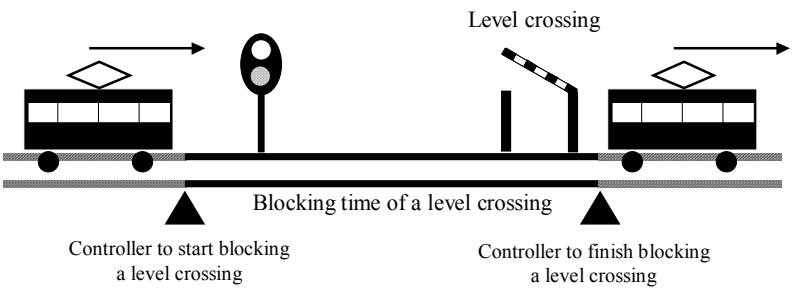
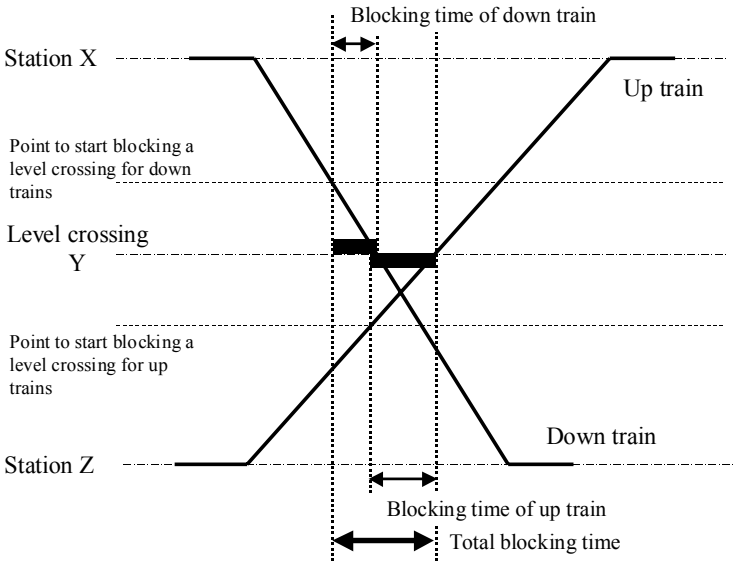


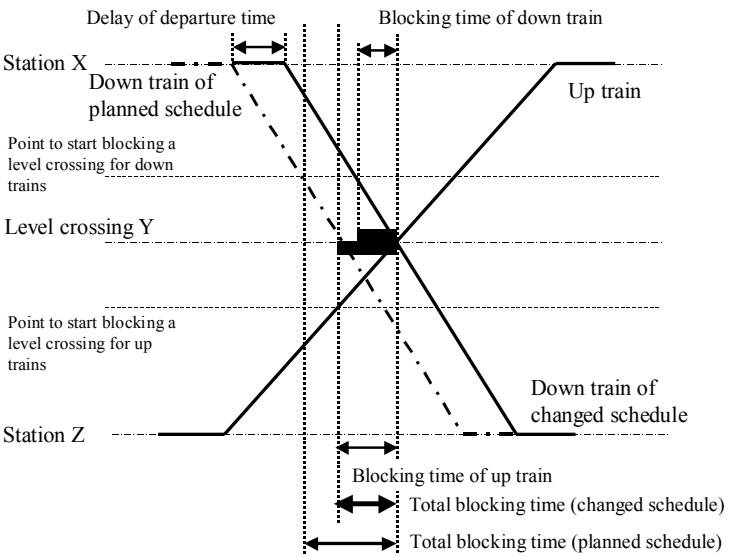
Figure 1: A block diagram of the control of a level crossing.

2.2 Outline of the proposed method and the application of GA

To calculate an optimal schedule that minimizes blocking time, we focus in this paper on a delay of the departure time at each station. As shown in Figure 2, two trains, an up train and a down train, pass through a level crossing in such a way as to maximize the time duration of the two trains being on the crossing simultaneously by delaying the departure time of the planned schedule, which results in a reduction of the blocking time of the level crossing.



(a) Planned schedule



(b) Changed schedule

Figure 2: A schematic diagram of the proposal method.

Here, for the purpose of securing the time for passengers to get on and off the train, the minimum standing time at each station was set at 20 seconds. To this we added a delay time for each train at each station in order to optimise the railroad schedule. Incidentally, because of this addition of a delay time and because of the fixed running time between stations, the arrival of a train at its terminal station will be delayed.

Now that we have shown the optimization method of a railroad schedule in view of reducing the blocking time of level crossings, we next examined the number of combinations of delay time. Because the departure time of each train at each station varies, the number of combinations in the proposed method is huge. For example, when the delay time is varied in five patterns (20[sec]+0[sec], +5[sec], +10[sec], +15[sec], and +20[sec]), and the number of up and down trains is each 20, and the number of stations is 10, then the number of combinations of departure time is 3.87×10^{279}. It is impossible to calculate all combinations in real time; therefore, we applied the genetic algorithm (GA), a powerful tool for the optimization of the problem of combinations, in our attempt to optimize the railroad schedule.

2.3 The model rail line and GA parameters

This time we used a model rail line that has 9 stations and 14 level crossings as shown in Figure 3 and Table 1. By using the delay time for each train at each station as a gene value of GA, we calculated the total blocking time of all level crossings per hour by means of the train simulator that we developed. This total blocking time was used as the fitness value.

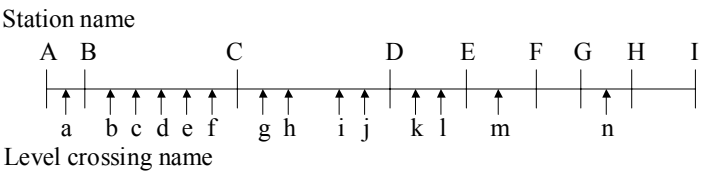


Figure 3: The model rail line.

Table 1: Distances on the model rail line.

(a) Stations

Name	Distance
A	0[km]
B	0.8[km]
C	1.7[km]
D	4.5[km]
E	7.2[km]
F	8.5[km]
G	9.7[km]
H	11.2[km]
I	13.4[km]

(b) Level crossings

Name	Distance	Name	Distance
a	0.221[km]	j	3.643[km]
b	0.934[km]	k	6.555[km]
c	1.084[km]	l	6.931[km]
d	1.232[km]	m	7.680[km]
e	1.420[km]	n	9.937[km]
f	1.651[km]		
g	1.867[km]		
h	2.386[km]		
i	3.415[km]		

3 Verification of effectiveness of the proposed method by the use of a computer simulation

To verify the effectiveness of the proposed method shown in the previous chapter, we carried out a computer simulation this time. Specifically, after we constructed railroad schedules on the model rail line with the running interval of the trains varying from 3 to 10 minutes, we developed the train simulator using GA to calculate the blocking time per hour of a level crossing in order to search for the combination of departure times that minimizes blocking time.

When we carried out the computer simulation in which the GA parameters were set so that the number of individuals was 30 and the number of generations was 1,000, we obtained the resultant characteristic of the total blocking time of all level crossings by changing the train intervals shown in Figure 4. Figure 4 affirms that we can reduce the total blocking time by changing the combinations of departure times. Especially, it shows a significant effect of reduction of the total blocking time by 2,500 seconds when we set the train interval to be 3 minutes with up and down trains crossing each other frequently. Moreover, under this condition, the delay of arrival time at the terminal station relative to the planned schedule was about 50 seconds on the average.

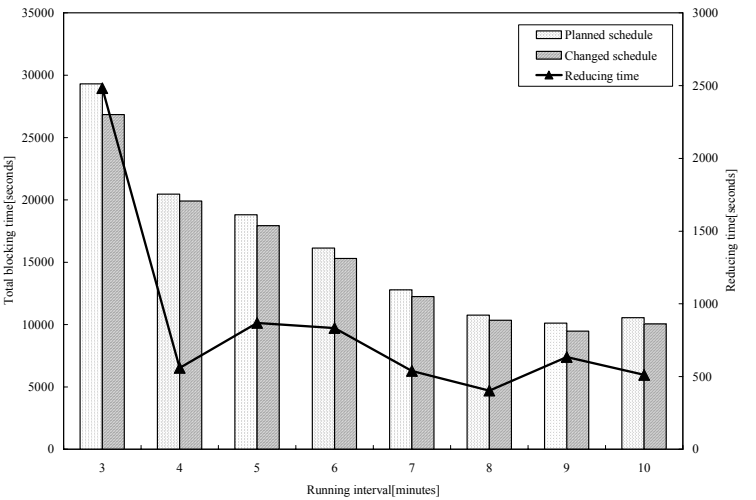


Figure 4: Characteristics of total blocking time by varying train intervals.

4 Examination of the composition of real-time calculation of optimal railroad schedule system using grid computing

4.1 Need for real-time calculation of the optimal railroad schedule

In the preceding chapter, we showed that we can reduce the total blocking time by changing the combinations of departure times. The time needed for

calculation is about 1 hour when the total number of individuals to be calculated is 30,000 (30 [individuals] x 1,000 [generations]). On the other hand, an actual railroad schedule on which trains run might be different from the planned schedule because the actual schedule is influenced by delays of departure due to passengers' movement, varying running times between stations due to weather, and so on. Based on this fact, we conclude that we must develop a way for real-time calculation of the optimal railroad schedule that can respond in real time to an actual running situation if we are to make practical use of this system. So we attempt to devise a way to speed up the present railroad schedule calculation.

4.2 Outline of the grid computing technique

To speed up the railroad schedule calculation, we used the grid computing technique, one of the distributed computing techniques. A grid computing system is composed of a master PC and some slave PCs, as shown in Figure 5. In actual processing, the master PC divides the processing that we want to be carried out into batches of a constant number of jobs, and the jobs are transferred to the slave PCs via the network. The slave PCs carry out their given jobs and return the results to the master PC. After repeating these operations, the master PC brings the processing results of the jobs together at the end. The grid computing technique offers some advantages, for example, an easy installation, low cost, and so on, so the grid computing technique is being used in various fields now.

4.3 Composition of real-time calculation system of the optimal railroad schedule

Based on the image chart that has been shown in Figure 5, we constructed a real-time calculation system of the optimal railroad schedule shown in Figure 6. In Figure 6, a GA engine plays the role of a master PC, and train simulators play the role of slave PCs. The actual processing goes as follows. A master PC that functions as the GA engine configures a constant number of departure time patterns for each station as initial genes and transfers them to slave PCs. The slave PCs that function as the train simulators calculate the total blocking time of all level crossings based on departure time patterns and return their results to the master PC. Then the master PC configures new departure time patterns by GA operation based on calculation results from the slave PCs and transfers them to slave PCs again. These operations are repeated.

4.4 Restructuring of GA engine to allow for the grid computing technique

We have shown the composition of the real-time calculation system of the optimal railroad schedule in Figure 6. Next we examined the restructuring of a GA engine to allow for grid computing. In a customary GA engine, after train simulators finished calculating the fitness values against a constant number of genes prepared in advance, the GA engine makes new genes by GA operation, and it searches for the optimal solution by repeating these operations. Thus, the

GA operation and fitness value calculation are done by turns. On the other hand, the purpose of the composition as shown in Figure 6 is to have both the master PC that functions as a GA engine and the slave PCs that function as the train simulators be active simultaneously. So it was necessary for us to restructure the GA engine for parallel processing. Specifically, we restructured the master PC to make new genes by GA operation based on the fitness values calculated by the slave PCs for part of the departure time patterns made as initial genes. By this means the slave PCs can carry out fitness value calculations for the rest of the genes while the master PC processes GA operation, and thus this composition enables us to have efficient parallel processing. Now that we have explained the restructuring of a GA engine, next we define the number of initial genes as the number of genes made by a GA engine in advance and the number of calculation genes as the number of genes used for GA operation, and we evaluated this composition thereafter.

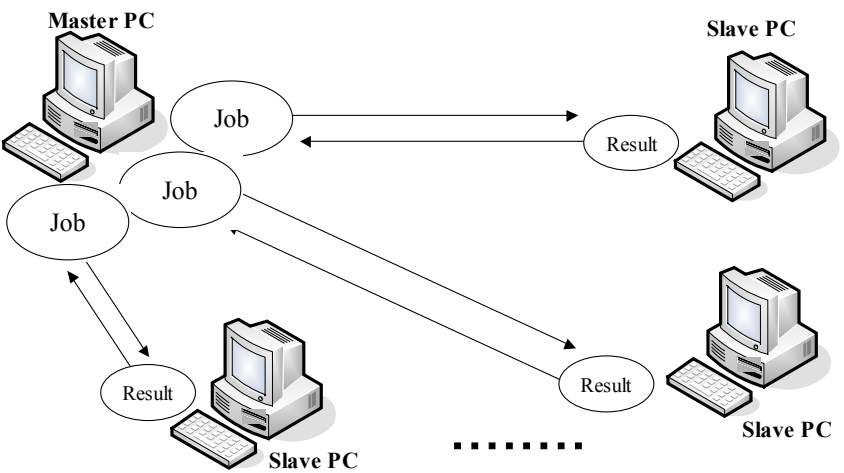


Figure 5: Image chart of grid computing.

4.5 Evaluation of a real-time calculation system of the optimal railroad schedule using a computer simulation

We evaluated the characteristics of the total blocking time when the number of calculation genes was fixed at 30, and the number of initial genes was varied in four patterns (30, 40, 59, and 60) by using a computer simulation. We show the result in Figure 7. As shown in Figure 7, no good characteristic can be obtained when the number of initial genes is 60, that is, twice the number of calculation genes, because this case is equivalent to evolution by two independent GA engines. On the other hand, good characteristics can be obtained when the number of initial genes is not 60. The characteristics when the number of initial

genes is 59 are evidently especially good compared with the characteristics of the traditional method (both the number of initial genes and the number of calculation genes are 30). In the above, we verified that this system can calculate the optimal railroad schedule most efficiently when the number of initial genes (N) is one less than twice the number of calculation genes (M). We call this the "N=2M-1 method" hereafter.

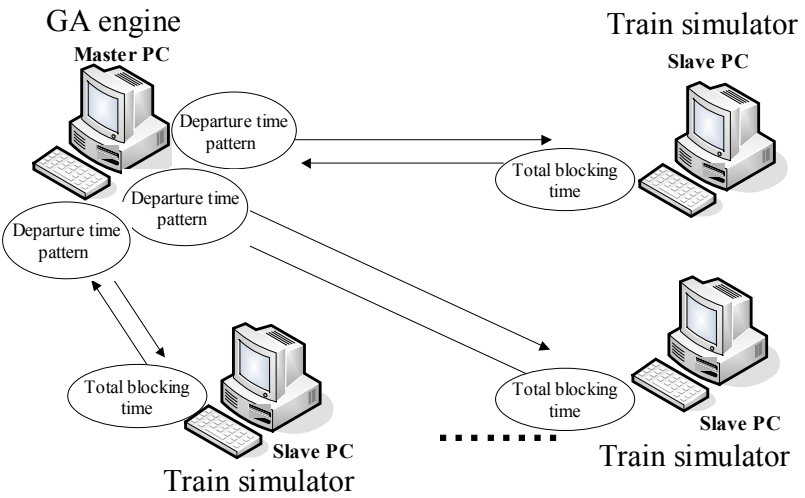


Figure 6: The composition of real-time calculation system of optimal railroad schedule.

Next, we examine the N=2M-1 method. We show the characteristics of the total blocking time when the number of calculation genes was varied in four patterns (15, 30, 60, and 120) in Figure 8. The numbers of initial genes for those four patterns were 29, 59, 119, and 239.

As shown in Figure 8, we verified that the characteristics of total blocking time for the number of departure time pattern calculations evolved quickly when the numbers of initial genes and calculation genes were small. And when the numbers were large, we verified that the characteristics evolved to a nearly optimal railroad schedule with a comparatively small number of calculations, even though the evolution speed was not fast. In fact, we took into consideration the characteristics as shown in Figure 8 when we mounted this system on grid computing. For example, when the number of initial genes was 120, the number of departure time calculations to optimize the railroad schedule was 15,000, which is one half of 30,000, the number needed in the traditional method. So if the number of slave PCs is 6, the optimal railroad schedule calculation time can be ideally shortened to 1/12 of the time needed in the traditional method. In sum, we have the prospects that the composition using the N=2M-1 method can calculate the optimal railroad schedule in a few minutes.

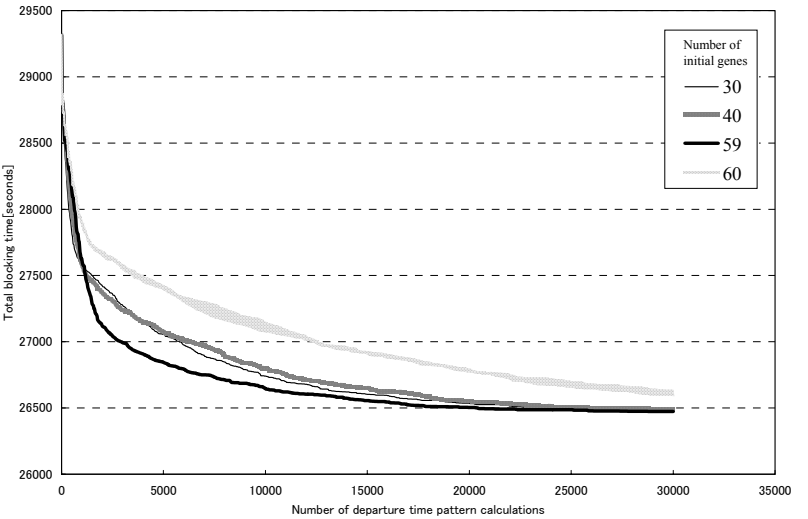


Figure 7: Characteristics of total blocking time when number of initial genes was changed.

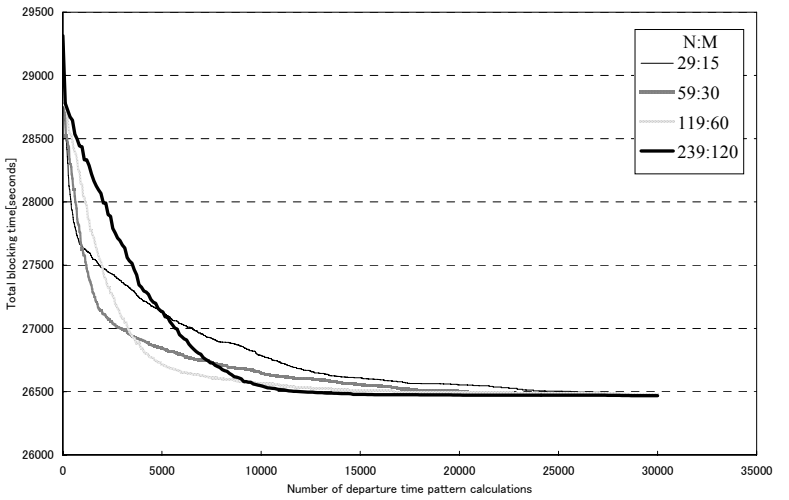


Figure 8: Characteristics of total blocking time in N=2M-1 method.

5 Conclusion

In this study, we examined a way to construct a railroad schedule in view of reducing blocking time of level crossings, and we proposed a method to calculate the pattern, using GA, that minimizes the total blocking time of all level

crossings by varying the time duration each train stands at each station. The result of a computer simulation verifies the effectiveness of the proposed method. In addition, we showed a composition of a real-time calculation system of the optimal railroad schedule using the grid computing technique that reduces calculation time. And as a result of restructuring the GA engine, we have the prospects that the railroad schedule should be optimized in a few minutes by the composition using the N=2M-1 method, in which we defined the relationship between the number of initial genes (N) and the number of calculation genes (M) as N=2M-1. In the future, we will implement the system that we proposed after we built a grid computing system actually using some PCs. And we hope to advance this research further by measuring the optimal railroad schedule calculation time and so on.

Reference

[1] Atsushi SAITO, Satoru SONE and Susumu TAKANO, "A Method of Controlling a Group of Trains for Securing Opening Time at Very Heavily Trafficked Double Track Railway Crossings", IEEJ, Papers of technical meeting, TER-03-14, pp.37-42 (2003-03).