# A Genetic Algorithm Approach for Solving the Train Formation Problem

DAVID MARTINELLI AND HUALIANG TENG

The train formation plan is one of the most important elements of railroad system operations. Although mathematical programming formulations and algorithms are available for solving the train formation problem (TFP), the computational time required for their convergence is usually excessive. At the same time, shorter decision intervals are becoming necessary given the highly competitive operating climates of the railroad industry. Thus, new techniques are needed for generating efficient solutions for the TFP. In this study, we present the development of a genetic algorithm (GA) as a possible technique for this problem. The calibration and validation of the GA model are carried out for three different complexity levels of objective functions. It is found that the optimal solutions can be found for all the different formulations while consuming only a small amount of computation time.

Railroad system operating plans are developed to perform the sequential decision process of: car block decisions, train formation decisions, train schedule decisions, and empty car distribution decisions. These are made under the consideration of engine power, maintenance, service level requirements, and other competing criteria. Car block decisions determine which blocks the cars will be assigned to, or which demand each block will carry. Train formation decisions determine which train the blocks will be assigned to, or which block each train will carry. Train schedule decisions determine when trains will be released from their origin station and arrive at their destination station. Finally, empty car distribution decisions determine where the empty cars will be sent. In this study, the train formation problem (TFP) is defined as: assign the traffic demand, in terms of cars, to available trains in a network environment so as to minimize the cost incurred in the whole production process.

Despite the substantial quantity and diversity of rail operating decision models, a common element exists in that they all require a substantial investment of computational effort and, subsequently, implementation time. Experience with these models indicate that the computational time required to obtain an optimal (or near optimal) solution varies with formulations.

A common approach for the industry in handling dynamic demands has been to shorten the time period between successive modeling updates. Unfortunately this introduces a tradeoff between longer central processing unit (CPU) time requirements for more realistic solutions and the added resources necessary to provide more frequent model updates. In light of this tradeoff, new approaches such as artificial intelligence are necessary and may prove quite fruitful if shorter implementation times can be achieved without a substantial loss in solution integrity (*1*). One such artificial intelligence technique is genetic algorithms (GA). In employing GA models, the intelligent optimal solution searching process alleviates the impacts of the inputs by directly going to the feasible solution region instead of stumbling on the restrictive constraints. This is the primary reason that genetic algorithms demonstrate promise as a solution technique for the TFP.

In comparison with conventional models, GA demonstrate several distinct advantages. First, they employ an efficient optimal solution searching technique which can be described as multi-hill climbing. The global solutions can be easily found for both linear and nonlinear formulations. Second, the optimal solution searching process is independent of the form of the objective function. Unlike conventional techniques, in which the algorithms usually rely on the structure of the formulation such as the conditions for the decomposition algorithms, GA models can be implemented without such considerations. Third, conventional algorithms are often sensitive to the input patterns such as the conditions set forth by Monte Carlo techniques.

There have been several transportation research efforts in which genetic algorithms are employed to deal with a combinatorial explosion associated with many optimization problems. Xiong and Schneider (*2*) integrated an artificial neural network model into a GA model to solve the traffic network design problem. Foy et al. (*3*) used a GA model to determine the optimal signal timing decisions in a simulation environment for on-line decision making. Chan et al. [unpublished data; cited by Xiong and Schneider (*2*)] applied a GA to road maintenance planning.

## BINARY INTEGER PROGRAMMING FORMULATIONS FOR THE TFP

An example railroad network having 6 nodes (representing yards) and 10 links (representing line segments) is represented in Figure 1. Trains are usually divided into long and short distance service. The short distance trains are those whose origin and destination yards are adjacent; whereas long distance trains are those whose origin and destination yards are not adjacent. Normally, short distance trains are always provided for each link, whereas the existence of long distance trains is determined by the train formation plan. Short distance demands are those whose origin and destination are connected directly by one link. Long distance demands are those whose origin and destination are not directly connected. In general, short distance demands are carried by short distance trains and long distance demands are carried by a combination of short and long distance trains.

In railroad networks, there are always a number of different physical routes available for a given demand. On a certain route, there are always a high number of possible itineraries (or assignments). These itineraries are distinguished from each other by the number

D. Martinelli, Department of Civil and Environmental Engineering, West Virginia University, P.O. Box 6103, Morgantown, W.Va. 26506. H. Teng, 1284 Civil Engineering Building, Purdue University, West Lafayette, Ind. 47906-1284.

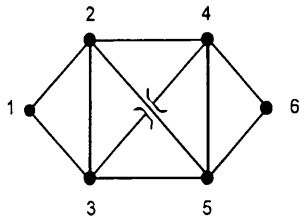


**FIGURE 1 Example railroad network.**

and types of trains. For example, for demand from Yard 1 to Yard 6, there might be four different physical routes possible: (1, 2, 4, 6), (1, 2, 5, 6), (1, 3, 4, 6), and (1, 3, 5, 6). Further, additional combinations exist for each route. For example, along physical route (1, 2, 4, 6), there might be four itineraries possible as represented in Figure 2. Referring to Itinerary $i_2$, the demand for Yard 1 to 6 will be relayed from Yard 1 to Yard 2, and then to Yard 6.

The designation of long distance trains and the route they follow are presented in Table 1. The corresponding demand matrix is represented in Table 2. The short distance trains are denoted such as T12 in Figure 2, where 1 and 2 are the train's origin and destination, respectively, whereas the long distance trains are in the form of T*ijk*, where *j* and *k* are the train's origin and destination, respectively, *i* is the sequence of the possible roads the train can follow between *j* and *k*.

It is a common practice for the sake of convenience that, when managing the traffic flow on the railroad network, each demand is usually confined to only one itinerary. If for each demand, a set of 0-1 variables are defined for the choice of itinerary, the TFP could be formulated as a 0-1 integer program. If the objective is minimizing the delay times including the travel times of the cars incurred in the railroad system, subject to demand routing deviation restriction as described above, then the TFP can be formulated as follows.

$$MIN \sum_{l=1}^{2L} t_l Y_l + \sum_{j=1}^{N} v_j x_j \tag{1}$$

Subject to:

$$\sum_{k \epsilon R_i} x_{i,k} = 1. \tag{2}$$

where:

$$x_j = \sum_{i=1}^{M} \sum_{k \epsilon s_j} r_i x_{i,k} \qquad Y_l = \sum_{i \epsilon p_l} x_i$$

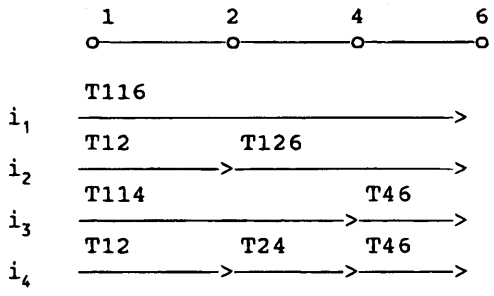


**FIGURE 2 Itinerary representation for route (1, 2, 4, 6).**

**TABLE 1 Designation for the Long Distance Train**

| Yard | Train | Train Route |
|---|---|---|
| 1 | T116 | (1,2,4,6) |
| | T216 | (1,2,5,6) |
| | T316 | (1,3,4,6) |
| | T416 | (1,3,5,6) |
| | T114 | (1,2,4) |
| | T214 | (1,3,4) |
| | T115 | (1,2,5) |
| | T215 | (1,3,5) |
| 2 | T126 | (2,4,6) |
| | T226 | (2,5,6) |
| 3 | T136 | (3,4,6) |
| | T236 | (3,5,6) |
| 4 | T141 | (4,2,1) |
| | T241 | (4,3,1) |
| 5 | T151 | (5,2,1) |
| | T251 | (5,3,1) |
| 6 | T161 | (6,4,2,1) |
| | T261 | (6,4,3,1) |
| | T361 | (6,5,2,1) |
| | T461 | (6,5,3,1) |
| | T162 | (6,4,2) |
| | T261 | (6,5,2) |
| | T163 | (6,4,3) |
| | T263 | (6,5,3) |

Here $X_j$ denotes the volume of cars in Train $j$, $Y_l$ the volume on Link $l$. Here, $t_l$ is the average travel time on Link $l$, and $v_j$ is Train $j$'s operating time at its destination yard. Also, $x_{i,k}$ is a binary integer variable representing the demand-itinerary choice. $x_{i,k}$ will be 1 if demand $i$ is carried by itinerary $k$, otherwise zero. $R_i$ is the amount of traffic of Demand $i$. $R_i$ denotes the set of itineraries by which Demand $i$ was supposed to be carried, $S_j$ is the set of itineraries which include Train $j$ as one part of their line haul, and $P_l$ the set of trains which pass through Link $l$. $L$ is the total number of links, $M$ the total number of demands, and $N$ the total number of trains possible provided. In this study, $L = 10$, $M = 30$ and $N = 44$. All the $t_l$s have values of 10 hr/car and the $v_j$s take values around 13–15 hr/car.

Equation 1 is the objective function, in which the first summation is for the travel times incurred on line segments, the second summation is for the times incurred at yards. Equation 2 is the demand-route restrictions. The demand flow conservation and balance constraints usually appear in transportation network models, but are automatically satisfied by this formulation. This is the first case we will investigate in this study.

**TABLE 2 Demand Matrix**

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | | 64 | 94 | 121 | 150 | 150 |
| 2 | 78 | | 87 | 27 | 54 | 107 |
| 3 | 72 | 95 | | 4 | 14 | 150 |
| 4 | 150 | 61 | 19 | | 10 | 34 |
| 5 | 136 | 38 | 89 | 87 | | 99 |
| 6 | 150 | 150 | 140 | 67 | 26 | |

In this constraint formulation, it is assumed that the times in which the traffic is incurred at yards and on line segments are independent of the traffic volume. However, in reality, the times are always dependent on the volume. The relationship between times and the traffic volume is nonlinear. Modifying the objective function accordingly, we have:

$$MIN \sum_{l=1}^{2L} t(Y_l)\, Y_l + \sum_{j=1}^{N} v(X_j)\, X_j \tag{3}$$

The formulation for this case is the objective function of Equation 4 plus the constraints in Case 1. This is denoted as Case 2 in this study.

Furthermore, in practice, it is likely to impose constraints on some variables such as link flow and train load. These constraints can be formulated as:

$Y_l \leq b_1$ for $l = 1, 2, \ldots, 2L$,
$X_l \geq b_7$ for $j = 1, 2, \ldots, N$.

The first indicates that the traffic volumes on links should be less than $b_1$. The second indicates that the trains can be provided only when the loads on them are larger than $b_2$. The formulation of Case 3 for this problem is that of Case 2 plus these two additional constraints.

Referring to the railroad networks in Figure 1, there are 10 long distance demands. For demand from 1 to 6 and from 6 to 1, each is assumed to have 16 possible itineraries. For the remaining 8 long distance demands, each is assumed to have 4 possible itineraries. For these conditions, the overall combinations of demands and itineraries is around $10^{17}$. All of the formulations in these three cases are binary integer programs. For Case 1, some algorithms such as branch-and-bound, cutting plane and Lagrangian relaxation have been proved to be effective conventionally. The common point of these algorithms might be the use of the linear characteristics of the objective function. In each operation of "branch," for example, relaxed linear programming can be efficiently solved. However, in Cases 2 and 3, the objective functions are not linear. To some extent, this makes the conventional approaches in Case 1 ineffective. Furthermore, these nonlinear functions are convex in nature. This makes the approximation approach almost impossible. With these difficulties, a GA approach is demonstrated in the following sections.

## INTRODUCTION TO GENETIC ALGORITHMS

GAs are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured, yet randomized, information exchange to form a search algorithm with some of the innovative flair of human search. In every generation, a new set of artificial creatures (strings) is created using bits and pieces of the fittest of the old. An occasional new part is tried for good measure. Although effective, GAs can be quite simple in their application, they efficiently exploit historical information to speculate on new search points with expected improved performance (*4*).

In the following section, the GA framework is introduced through a simple optimization problem (SOP):

$$\text{MAX} \quad 1024 - (x - 31)^2 \tag{4}$$

**TABLE 3 Solution Strings and Fitness Values**

| No. | x | String | Fitness | % of Total | Expected Count |
|---|---|---|---|---|---|
| 1 | 5 | 000101 | 348 | 9.4 | 0.566 |
| 2 | 42 | 101010 | 903 | 24.5 | 1.469 |
| 3 | 53 | 110101 | 540 | 14.6 | 0.878 |
| 4 | 38 | 100110 | 975 | 26.4 | 1.586 |
| 5 | 61 | 111101 | 124 | 3.4 | 0.202 |
| 6 | 16 | 010000 | 799 | 21.7 | 1.300 |
| Total | | | 3689 | 100 | |
| Average | | | 614.8 | | |

*Subject to* $0 \leq x \leq 63$ (5)

where $x$ is an integer.

First, The bit string representation is implemented by the widely used list of 0's and 1's. Table 3 shows the binary strings for six solutions: 5, 42, 53, 38, 61, and 16. The evaluation function is the same as Equation 5, and fitness values for the six solutions are also listed in Table 3. Second, these six solutions are assumed to be the first generation. Third, to generate the offspring generation, three commonly used operators are employed: reproduction, crossover, and mutation. These three operators are applied, in turn, to the solutions in the current generation during the search process.

The first operator, reproduction, is a process in which good solutions survive and are retained and bad solutions die. The number of solutions reproduced by each original solution is proportional to its fitness value. For instance, referring to Table 3, String 1 has a fitness of 348, which represents 9.4 percent of total fitness of the population of solutions. Therefore, its expected count is 0.566 which is obtained through dividing its fitness by the average fitness. Hence, using the population shown in Table 3 as parents, a possible population generated by reproductions is shown in Table 4, in which String 4 in Table 3 produces two solutions whereas String 5 produces none.

The second operator, crossover, is generally performed on the population newly generated by reproduction. The crossover proceeds in two steps. First, members of the reproduced strings are mated at random. Second, the two solutions in each solution pair exchange their "chromosomes" which are represented by an alphabetic string. Suppose in Table 4, String 1 is mated with String 2, String 3 with String 4, and String 5 with String 6. If $k$ is 1, 4, and 2 for these three pairs, respectively, the population generated will be as shown in Table 5, in which the crossover sites are denoted by "|."

After crossover, a mutation operator is used. This operation works on a bit-by-bit basis. It simply changes every bit (or character) in every solution string in the population to its opposite bit

**TABLE 4 Population Generated by Reproduction**

| No. | x | String | Fitness |
|---|---|---|---|
| 1 | 5 | 000101 | 348 |
| 2 | 42 | 101010 | 903 |
| 3 | 53 | 110101 | 540 |
| 4 | 38 | 100110 | 975 |
| 5 | 61 | 111101 | 124 |
| 6 | 16 | 010000 | 799 |

**TABLE 5 Population Generated by Crossover**

| No. | Old String | Mate | New String | x | Fitness |
|---|---|---|---|---|---|
| 1 | 0\|00101 | 1 | 001010 | 10 | 583 |
| 2 | 1\|01010 | 1 | 100101 | 37 | 988 |
| 3 | 1101\|01 | 2 | 110110 | 54 | 495 |
| 4 | 1001\|10 | 2 | 100101 | 37 | 988 |
| 5 | 10\|0110 | 3 | 100000 | 32 | 1023 |
| 6 | 01\|0000 | 3 | 010110 | 22 | 943 |
| Total | | | | | 5020 |
| Average | | | | | 836.7 |

(or other character) with a very small probability. Suppose that, in this SOP, the mutation probability is .001. In this particular case, it is very likely that no mutation will be made to any bit and the resulting population will not differ from that shown in Table 5.

Finally, after the operations of reproduction, crossover, and mutation, the population of a new generation becomes those presented in Table 5. The average fitness value for the SOP has been increased from 614.8 in Table 3 to 836.7 in Table 5, and the maximum fitness has also increased from 975 to 1023, respectively.

## GA FORMULATION TO THE TFP

Referring first to the conventional formulation, train formation decisions are represented by 0-1 strings which are illustrated in Figure 3. In this figure, $x_{i,k}$ is the decision variable from Equation 1 where each route for each demand has two itineraries.

Second, the evaluation functions derived for the three cases are the following:

$$BM - \left[\sum_{l=1}^{2L} t(Y_l)\, Y_l + \sum_{j=1}^{N} v(X_j)\, X_j\right] \tag{6}$$

$$BM - \left[\sum_{l=1}^{2L} t_l Y_l + \sum_{j=1}^{N} v_j X_j\right] \tag{7}$$

$$BM - \left[\sum_{l=1}^{2L} t(Y_l)\, Y_l + \sum_{j=1}^{N} v(X_j)\, X_j + \sum_{l=1}^{2L} f(Y_l - b_l) + \sum_{j=1}^{N} g(X_l - b_2)\right] \tag{8}$$

where $BM$ is used to convert the minimizing objective functions to maximizing. The variables $f$ and $g$ are penalty functions.

Third, the GA operations are designed as follows: 1) The initial set of solutions are generated randomly and 2) The operations of reproduction and mutation are the same as those demonstrated in the SOP. However, with regard to the constraints represented in Equation 2, the mate sites in the crossover operation are selected uniformly at random from a specific set of positions, instead of from a set of consecutive numbers like that in the SOP. In this way, the constraint in Equation 2 can be guaranteed automatically in the genetic algorithm operation.

## CALIBRATION OF THE GA MODEL

The calibration process for the GA model is to find the appropriate parameters by which the best solutions of the GA model can be obtained. These parameters include the size of the population, the number of generations, the crossover probability, and the mutation probability. In order to quicken the calibration, it is decomposed into two steps. The first step considers only the first two parameters. When generating the schemes, only the first two parameters vary within certain ranges, whereas the last two parameters are fixed at 0.9 and 0.03, respectively. From this step, the optimal number of generations and population size are determined. Given these determined values, the second step generates schemes by varying the last two parameters in a certain range. From this step, the optimal values for crossover and mutation probability are obtained. This process is conducted for all three cases. The details of the validation are described as follows.

In the first step, the generations are set at 100, 200, . . . , 1000, respectively. The population sizes are set at 10, 20, . . . , 100, respectively. Then, for each case, 100 schemes will need to be generated and evaluated. After a rough scanning of all the results, it is determined that the generation of 1000 is the most appropriate to evaluate the performance of the GA model. Then, the remaining task is to investigate the influence of the population size on the search process. The results are plotted in Figure 4 for Case 1. Similar plots were generated and used for Cases 2 and 3. The population sizes are determined by two criteria: the time the GA model uses to decrease the objective function values to the best solution and the stability after the best solutions have been achieved. For some processes, the convergence from the initial objective function value to the optimum is rather quick. On the other hand, other processes will fluctuate around the optimum value. The optimal population size is found to be 10 for Case 1, 100 for Case 2, and 70 for Case 3.

Following the procedures for Step 2, the results listed in Table 6 are obtained. The crossover probabilities are set at 0.6, 0.7, 0.8, 0.9, and 1.0 respectively. The mutation probabilities are set at 0.01, 0.02, 0.03, 0.04, and 0.05, respectively. For ease of analysis, the solution

| Demand i | | | | Demand i+1 | | | |
|---|---|---|---|---|---|---|---|
| Route 1 | | Route 2 | | Route 1 | | Route 2 | |
| itinerary k | itinerary k+1 | itinerary k+2 | itinerary k+3 | itinerary k+4 | itinerary k+5 | itinerary k+6 | itinerary k+7 |
| $x_{i,k}$ | $x_{i,k+1}$ | $x_{i,k+2}$ | $x_{i,k+3}$ | $x_{i+1,k+4}$ | $x_{i+1,k+5}$ | $x_{i+1,k+6}$ | $x_{i+1,k+7}$ |

**FIGURE 3 GA string representation of train formation decision.**

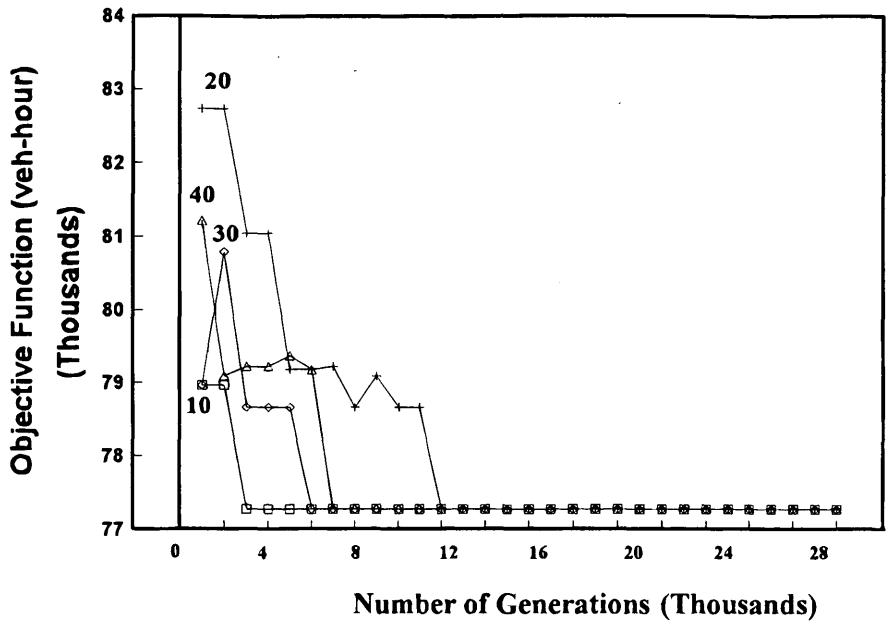


FIGURE 4 Determination of population size for Case 1.

TABLE 6 Calibration for Crossover and Mutation Probability in Cases 1, 2, and 3

| Mutation Probability | Crossover Probability Case 1 | | | | |
|---|---|---|---|---|---|
| | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
| 0.01 | 3 | 2 | 2 | 2 | 2 |
| 0.02 | 3 | 2 | 2 | 2 | 2 |
| 0.03 | 4 | 3 | 3 | 3 | 3 |
| 0.04 | 4 | 2 | 3 | 3 | 3 |
| 0.05 | 8 | 2 | 4 | 4 | 10 |
| **Mutation Probability** | **Crossover Probability Case 2** | | | | |
| | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
| 0.01 | 19 | 10 | 16 | 16 | (3) |
| 0.02 | 46 | (3) | 186 | (2) | 19 |
| 0.03 | (2) | 160 | 73 | (2) | (3) |
| 0.04 | 853 | 117 | 73 | 48 | 81 |
| 0.05 | (3) | (4) | (2) | (3) | (4) |
| **Mutation Probability** | **Crossover Probability Case 3** | | | | |
| | 0.6 | 0.7 | 0.8 | 0.9 | 1.0 |
| 0.01 | (3) | (2) | 46 | (3) | (3) |
| 0.02 | (4) | 92 | 32 | (3) | (3) |
| 0.03 | 72 | 43 | 277 | 529 | 834 |
| 0.04 | (2) | 121 | (2) | 123 | (2) |
| 0.05 | (3) | (4) | (2) | (2) | (2) |

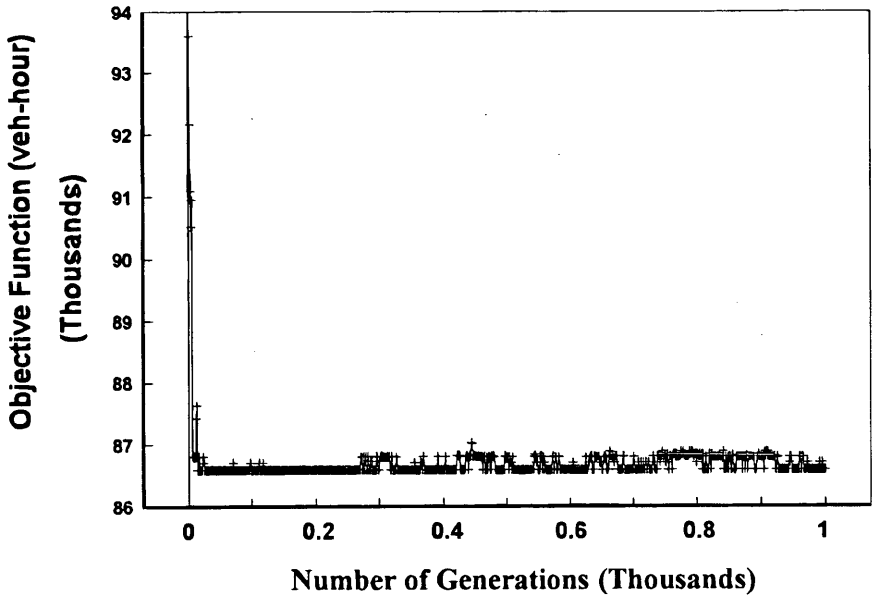


**FIGURE 5 Search process: Pattern 2.**

searching processes are classified into four patterns. In Pattern 1, the searching processes are stable after the smallest values are found. The generation at which the smallest values are found is called the stable generation. This pattern is viewed to have the best performance. Pattern 2 and Pattern 3, which are represented in Figures 5 and 6, respectively, are similar in the solution search processes. Both patterns indicate that the search processes will fluctuate after the smallest objective function value is achieved. However, in Pattern 2, the search process stays at the convergence status for a longer time than that in Pattern 3. Further comparing with Pattern 4, which is represented in Figure 7, the extent of fluctuation in Pattern 2 and 3 is smaller than that in Pattern 4. Among these four patterns, Pattern 1 shows a strong ability to keep the smallest values they achieved. Pattern 4 is the worst condition.

Referring to Table 6, the number outside parentheses represents the stable generation in Pattern 1, and the numbers in parentheses represent the designation of patterns. The crossover and mutation probabilities are determined by the corresponding row and column values of the cell which have the smallest stable generation. In Case 1, the crossover and mutation probabilities are determined to be .7 and .01, respectively. For Case 2, corresponding the stable generation of 10, they are determined to be .7 and .01, respectively. For the Case 3, corresponding to the stable generation of 32, they are determined to be .8 and .02, respectively.

Referring to Table 6, it can be seen that the linear case (Case 1) involves fewer generations to obtain the optimal solution than the nonlinear cases (Case 2 and Case 3). In Case 1, regardless of the parameters, the GA model always obtains the optimal solutions. In

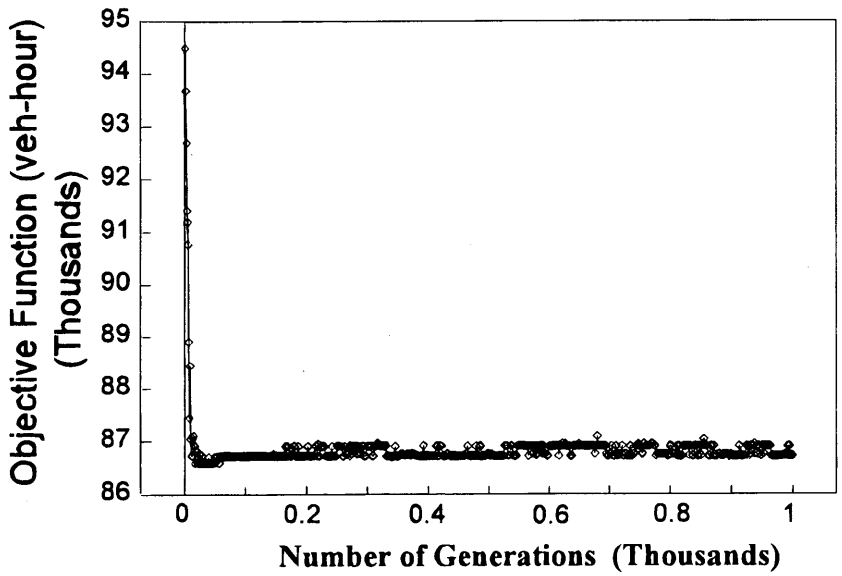


**FIGURE 6 Search process: Pattern 3.**

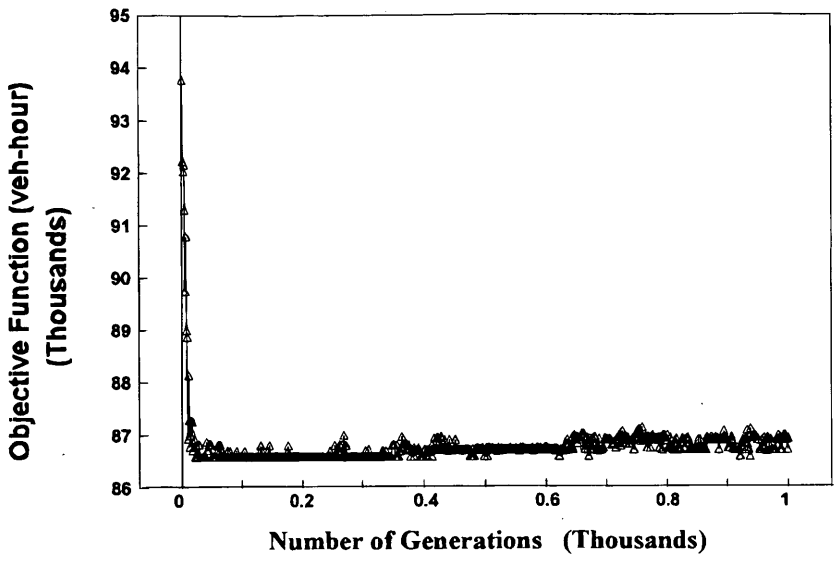


**FIGURE 7 Search process: Pattern 4.**

Case 2, there are 11 schemes which cannot yield the optimal solution; in Case 3, there are 15 such schemes. Furthermore, from Table 6, it can be seen that additional constraints in Case 3 make the solution searching process longer. In Case 2, there are 14 stable schemes, and the average stable generation is 123, whereas in Case 3, there are 10 such schemes with an average of 217.

## VALIDATION OF THE GA MODEL

The task of the validation process of the genetic algorithm model is to test whether the obtained solutions are optimal. In this study, different validation methods are employed for different cases. For Case 1, because the optimal solutions can be obtained from available software developed by conventional algorithms, the solutions from the GA model are compared with them. For Cases 2 and 3, the optimality of the solutions from the GA model are evaluated by observing whether the solutions have obeyed the constraints imposed by the nonlinear delay time functions and imposed by some practical considerations. After the feasibility of the solutions is determined, a variety of combinations around the obtained solutions are tested to see whether smaller objective values can be achieved.

In Case 1, the optimal solution is obtained by using Quant Systems (Version 2.1). This solution is compared with that obtained from the GA model. The solutions are the same, except that GA models can also produce other optimal solutions when multiple solutions exist. In each case, all the solutions are optimal, whereas the itinerary choices are different. The train loads and link volumes from one of the optimal solutions are listed in Tables 7 and 8.

In Case 2, the link volumes should be as small as possible, because convex links and train performance functions are used. From the point of view of networks, however, the link volumes should not fluctuate dramatically. Observe Table 8, comparing with the solution of Case 1, the link volumes in Case 2 are indeed evenly distributed. The largest value in Case 1, 508, disappeared in Case 2. Referring to Figure 3, the load on the trains should be as much as possible. In Table 7, all of the trains are assigned as large a load as possible. From these two observations, it is apparent that the solutions are feasible and the nonlinear functions are effective. Following this analysis, the solutions are analyzed by providing all possible and comparable schemes. It is concluded that the solutions are truly optimal in terms of minimizing the total delay in the railroad system.

In Case 3, beside the nonlinear objective function, the remaining two constraints are added, that is, the volume on each link should

**TABLE 7 Train Loads of the Three Cases for Validation**

| | The Load Assigned for the Long Distance Train | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | T114 | T214 | T115 | T215 | T116 | T216 | T316 | T416 | T126 | T226 | T136 | T236 |
| Case 1 | 121 | 0 | 0 | 150 | 150 | 0 | 0 | 0 | 0 | 107 | 150 | 0 |
| Case 2 | 121 | 0 | 150 | 0 | 0 | 0 | 150 | 0 | 107 | 0 | 0 | 150 |
| Case 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | The Load Assigned for the Long Distance Train | | | | | | | | | | | |
| | T141 | T241 | T151 | T251 | T161 | T261 | T361 | T461 | T162 | T262 | T163 | T263 |
| Case 1 | 0 | 150 | 0 | 136 | 0 | 0 | 0 | 150 | 150 | 0 | 140 | 0 |
| Case 2 | 0 | 150 | 136 | 0 | 150 | 0 | 0 | 0 | 0 | 150 | 0 | 140 |
| Case 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**TABLE 8 Link Volumes of the Three Cases for Validation**

| | Traffic Volume on Each Link of Each Direction (car) | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | link 12 | link 21 | link 13 | link 31 | link 24 | link 42 | link 35 | link 53 | link 25 | link 52 |
| Case 1 | 335 | 78 | 244 | 508 | 298 | 211 | 164 | 375 | 161 | 38 |
| Case 2 | 335 | 364 | 244 | 222 | 255 | 211 | 164 | 229 | 204 | 324 |
| Case 3 | 335 | 364 | 244 | 222 | 255 | 211 | 164 | 229 | 204 | 324 |
| | Traffic Volume on Each Link of Each Direction (car) | | | | | | | | | |
| | link 34 | link 43 | link 46 | link 64 | link 56 | link 65 | link 23 | link 32 | link 45 | link 54 |
| Case 1 | 154 | 319 | 334 | 357 | 206 | 176 | 87 | 95 | 10 | 82 |
| Case 2 | 154 | 169 | 291 | 217 | 249 | 316 | 87 | 95 | 10 | 82 |
| Case 3 | 154 | 169 | 291 | 217 | 249 | 316 | 87 | 95 | 10 | 82 |

be no more than 300 cars (which is realized by setting $b_1$ equals 300), and the load on each long distance train should be more than 200 cars (which is realized by setting $b_2$ equals 200). These two constraints establish that the possible load on each of the long distance trains are not sufficiently large to justify the provision. In Table 7, the loads are really zero, whereas in Table 8, it appears that the link volume constraints are not effective. However, after careful calculation, the overall demand through those links where the volumes exceed 300, it is found that there is no way to distribute these volumes without avoiding the penalty of the violation of the constraints. Thus, the solution is truly the optimal.

## GA MODEL COMPUTATIONAL PERFORMANCE

In Case 1, the Quant Systems consumes 1.17 sec of CPU to produce the optimal solution. However, the GA model uses less time.

In Cases 2 and 3, for the number of generations equal to 1000, the GA model requires approximately 10 min of CPU. Because both cases can obtain the optimal solutions in less than 40 generations, the computation time should be about 20 sec. Comparing with the size of the problem ($10^{17}$), this computation time is quite satisfactory.

Using the calibrated parameters, the GA model is used for varieties of demand patterns. In Table 2, the long distance demands are varied in the range of 100 to 150 cars; there are almost no computation time variations. For all the demand patterns, the GA model produces the optimal solutions within 40 generations.

## CONCLUSIONS

Several conclusions can be derived from this study. First, a GA model is able to produce optimal solutions for the formulations which might be difficult conventionally. Also, the computation time is satisfactory. Second, a GA model is not as sensitive to the input patterns compared to Monte Carlo algorithms. Third, the implementation process for a GA model is straight forward. In all three cases, the implementation simply involves the adjustment of the objective function formulations. There is no need to give the structure of the formulation a special consideration. The calibration and validation process are also straight forward. Fourth, the binary representation for the binary integer program (BIP) is especially effective.

Based on the principle introduced in this study, GA models can likely be effective when applied to large railroad networks. The patterns recognition of the solution searching process needs to be analyzed quantitatively instead of qualitatively, however. To this end, some statistical model might need to be developed.

## ACKNOWLEDGMENT

The authors wish to thank Lijuan Su, who assisted in the data generation and processing tasks of this research.

## REFERENCES

1. Martinelli, D. R., and H. Teng. A Neural Network Approach for Solving the Train Formation Problem. Presented at 73rd Annual Meeting of the Transportation Research Board, Washington, D.C., 1994.
2. Xiong, Y., and J. B. Schneider. Transportation Network Design Using a Cumulative Genetic Algorithm and a Neural Network. Presented at 71st Annual Meeting of the Transportation Research Board, Washington, D.C., 1992.
3. Foy, M. D., R. F. Benekohal, and D. E. Goldberg. Signal Timing Determination Using Genetic Algorithms. In *Transportation Research Record 1365*, TRB, National Research Council, Washington, D.C., 1993, pp. 108–115.
4. Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Mass., 1987.

*Publication of this paper sponsored by Committee on Artificial Intelligence.*