Optimal Programming by Genetic Algorithms for Pavement Management

T. F. FWA, W. T. CHAN, AND C. Y. TAN

Optimal programming of pavement activities is a desired element in pavement management systems. The complexity and scale of the problem, however, have prevented the widespread use of such analytical tools in practice. An application of a relatively new optimization technique, known as the *genetic algorithms,* to the pavement programming problem is described. This technique does not require any information on the differentiability, convexity, or other auxiliary properties of the problem parameters. On the basis of the mechanics of natural selection it has a robust search algorithm that makes it an attractive technique for pavement programming at the network level. Through an example application to a network-level pavement programming problem, the considerations involved in genetic representation of the problem and generation of new solutions (known as *offspring*) are presented. The convergence characteristics of the genetic algorithm solutions are also analyzed. Finally, the applicability of the technique to the general pavement management problem is discussed.

A primary objective of pavement management at the network level is to program pavement investments and schedule pavement activities to achieve optimal results of pavement network performance. Many optimization techniques have been developed since the mid-1970s to provide the necessary analytical tools to assist highway agencies in making such management decisions. These techniques include dynamic programming (*1*), optimal control theory (*2*), integer programming (*3*), linear programming (*4*), and nonlinear programming and heuristic methods (*5*). Because of the complexity of the pavement programming problem at the network level, different techniques are suitable under different circumstances.

This paper illustrates the application of a general purpose problem-solving and optimization technique, known as the *genetic algorithms,* (GAs) to the pavement management problem. The genetic algorithms are formulated loosely on the basis of the principles of Darwinian evolution (*6,7*). The general operating principles of genetic algorithms are presented first in this paper; this is followed by an application example that solves a network-level pavement programming problem.

OPERATING PRINCIPLES OF GAs

Theoretical Basis of GAs

GAs are robust search techniques formulated on the basis of the mechanics of natural selection and natural genetics. It was in the 1980s that genetic algorithm applications started to spread across a broad range of disciplines, including function optimizers (*8*), pattern recognition (*9,10*), computer-aided operation control (*11*), and robot kinetics (*12*).

GAs are different from traditional optimization techniques in a few important aspects. First, GAs work with a coding of the parameters instead of the parameters themselves. The choice of the parameter representation is important because GAs work on a coded version of the problem to be solved and not on the problem directly. Second, GAs operate by manipulating a pool of feasible solutions instead of one single solution in the search of good solutions. Working with a pool of solutions enables GAs to identify and explore properties that good solutions have in common. Third, GAs employ probabilistic transition rules to move from one pool of solutions to another. This introduces perturbations to move out of local optima. Last, GAs rely only on objective function evaluations. They do not require any information on differentiability, convexity, or other auxiliary properties. GAs are thus easy to use and implement for a wide variety of optimization problems.

Mechanics of GA Solution Process

For a given problem with a specified objective function, the problem-solving process of GAs begins with the identification of problem parameters and the genetic representation (i.e., coding) of these parameters. The search process of GAs for solutions that best satisfy the objective function involves generating an initial random pool of feasible solutions to form a parent solution pool and obtaining new solutions and forming new parent pools through an iterative process of copying, exchanging, and modifying parts of the genetic representations in a fashion similar to that of natural genetic evolution.

Each solution in the parent pool is evaluated by means of the objective function. The fitness value of each solution, as given by its objective function value, is used to determine its probable contribution in the generation of new solutions, known as *offspring*. The next parent pool is then formed by selecting the fittest offspring on the basis of their fitnesses (i.e., their objective function values). The entire process is repeated until a predetermined stopping criterion is reached on the basis of either the number of iterations or the magnitude of improvement in the solutions. Figure 1 presents a flow chart that summarizes this solution process.

EXAMPLE PROBLEM

Problem Description

The application of GAs to pavement management is illustrated in this paper by solving an integer-programming optimization problem

Center for Transportation Research and Department of Civil Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 0511.

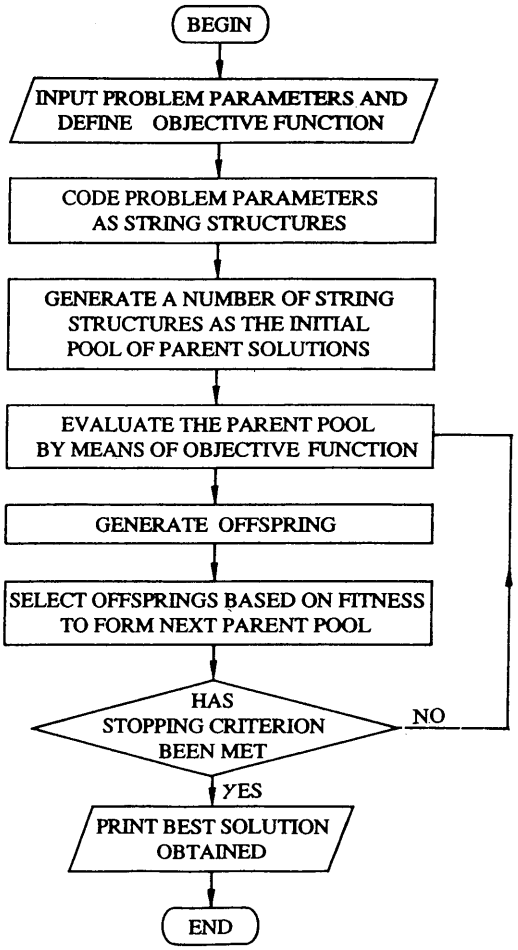

FIGURE 1 Solution process of GAs.

analyzed by Fwa et al. (*3*). On the basis of the framework of pavement management practice in Indiana, Fwa et al. solved for an optimal pavement repair program at the network level for a given rehabilitation schedule and subject to six forms of resources and operation constraints. Mathematically, the problem can be expressed as follows:

$$\text{Maximize} \sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3} W_{ijk}F_{ijk} \tag{1}$$

with W_{ijk} as integers for $i = 1, 2, \ldots N_1$, $j = 1, 2, \ldots N_2$, $k = 1, 2, \ldots N_3$, subject to the following constraints:

1. Production requirements:

$$0 \le W_{ijk} \le \frac{T_{ijk}\gamma_{ijk}}{U_{ijk}} \qquad i = 1, 2, \ldots N_1, j = 1, 2, \ldots N_2, \quad k = 1, 2, \ldots N_3, \tag{2}$$

2. Budget constraint

$$\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3} W_{ijk}U_{ijk}C_{ijk} \le B \tag{3a}$$

$$\sum_{i=1}^{N_1}\sum_{k=1}^{N_3} W_{ijk}U_{ijk}C_{ijk} \le b_j \qquad j = 1, 2, \ldots N_2 \tag{3b}$$

3. Manpower availability

$$\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3} W_{ijk}h_{jg} \le H_g \qquad g = 1, 2, \ldots, G \tag{4}$$

4. Equipment availability

$$\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3} W_{ijk}q_{jr} \le Q_r \qquad r = 1, 2, \ldots, R \tag{5}$$

5. Material availability

$$\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}\sum_{k=1}^{N_3} W_{ijk}m_{js} \le M_s \qquad s = 1, 2, \ldots, S \tag{6}$$

6. Rehabilitation constraints

$$\gamma_{ijk} = \frac{D - d_{ijk}}{D} \qquad i = 1, 2, \ldots N_1, j = 1, 2, \ldots N_2, \quad k = 1, 2, \ldots N_3, \tag{7}$$

where all variables are defined in Table 1. The practical meaning and rationale for each of the above equations are found in the technical paper by Fwa et al. (*3*). Fwa et al. solved this problem by the integer programming technique with the branch and bound algorithm of the multipurpose optimization scheme (MPOS) (*13*).

Input Data

The problem solved by Fwa et al. (*3*) considered four pavement defects and three levels of maintenance-need urgency. Table 2(*a*) gives the production rate and unit cost data for each combination of pavement defect and urgency level. The requirements for four manpower types and six equipment types are listed in Table 2(*b*). Recorded in Table 2(*c*) are the pavement repair priority weighting factors, which are functions of highway class, pavement repair activity type, and its need-urgency level.

The estimates of the amount of work required for each type of repair in terms of workdays are found in Table 3(*a*). The input for rehabilitation constraint factors are given in Table 3(*b*). A zero value of γ_{ijk} represents a case in which there is a complete interference from rehabilitation work, whereas a γ_{ijk} value of unity indicates no interference from rehabilitation. Other necessary input information, namely, the analysis period, budget allocation, manpower availability, and equipment availability, are found in Table 3(*c*).

APPLICATION OF GAs TO EXAMPLE PROBLEM

Main Features of Problem

Two main features of the problem affect the choice of solution techniques. First, the decision variables W_{ijk} are integers that restrict the methods to those that can handle integer variables. The other feature of the problem is what is commonly known as the *combinatorial explosion* of the feasible solution space. In the current problem, there are four highway types ($i = 4$), four pavement repair activities ($j = 4$), and three need-urgency levels ($k = 3$). There are altogether 48 ($4 \times 4 \times 3$) decision variables (W_{ijk}). In the extreme case in which each decision variable is allowed to take up any integer value from 0 to 45 workdays, the total number of possible solutions is equal to 46^{48}, or 6.4×10^{79}. Even if one were to assume a case in

TABLE 1 Definitions of Variables in Equations 1–7

Variable	Definition
W_{ijk}	equivalent workload units in number of workdays of pavement repair activity j of need urgency level k performed on highway i
F_{ijk}	priority weighting factor for pavement repair activity j of need urgency level k on highway i
N_1	total number of highways considered
N_2	total number of pavement repair activities considered
N_3	total number of need urgency levels considered
T_{ijk}	total workload of pavement repair needs expressed in work measurement units (see Table 2) for pavement repair activity j of need urgency level k on highway i
γ_{ijk}	rehabilitation constraint factor for pavement repair activity j of need urgency level k, $0 \leq \gamma_{ijk} \leq 1$
U_{ijk}	work productivity for pavement repair activity j of need urgency level k on highway i
C_{ijk}	cost per production unit of pavement repair activity j of need urgency level k on highway i
B	total budget amount allocated for the analysis period considered
b_j	budgeted fund for pavement repair activity j
h_{jg}	number of mandays of work crew type g required for each unit of pavement repair activity j
H_g	total available number of mandays of work crew type g
G	total number of work crew type
q_{jr}	number of equipment days of equipment type r required for each production day of pavement repair activity j
Q_r	total available number of equipment days of equipment type r
R	total number of equipment types
m_{js}	quantity of material type s required for each production day of pavement repair activity j
M_s	total available quantity of material type s
S	total number of material types
d_{ijk}	number of working days before a scheduled rehabilitation during which no pavement repair activity j of need urgency level k would be performed on highway i
D	total number of working days in analysis period

which each decision variable could only take up a value from 0 to 5 workdays, the total number of possible solutions of 6^{48}, or 2.2×10^{37}, would still require a modern supercomputer many years to enumerate all possible solutions.

Genetic Representation of Problem

In GAs a solution to a problem is represented by a string structure similar to the chromosomes in natural evolution. As shown in Figure 2 the chromosomal representation of a solution is known as a *genotype,* which consists of a string of genes. The value of each gene is called its *allele.*

Each genotype is a solution in the structure of the solution space represented by the genetic representation chosen. For example, Figure 3 shows a genotype that represents a solution with the values of W_{ijk} indicated therein. Because each W_{ijk} can assume any integer value from 0 to 45, the representation is different from the traditional binary representation used in GA applications.

GA Operations

After the genetic representation is determined, an initial parent pool of solutions can be randomly generated. A pool of 80 solutions was selected for the present example. The following three GA operations were then executed in sequence repeatedly: (a) generation of offspring, (b) the formation of of a new parent pool, and (c) performance evaluation and convergence assessment of the parent pool solutions.

TABLE 2 Production and Resource Requirements Data

(a) Production rate and unit cost data

Need-Urgency Level k	Production Rate U_{ijk}				Unit Cost C_{ijk}			
	Shallow Patching j = 1 (kg mix per day)	Deep Patching j = 2 (kg mix per day)	Premix Leveling j = 3 (kg mix per day)	Crack Sealing j = 4 (km per day)	Shallow Patching j = 1 ($ per kg mix)	Deep Patching j = 2 ($ per kg mix)	Premix Leveling j = 3 ($ per kg mix)	Crack Sealing j = 4 ($ per km)
High (k=1)	6,537.6	17,978.4	10,896.0	10.1	0.0938	0.0852	0.0403	81.37
Medium (k=2)	3,813.6	9,443.2	80,448.8	13.5	0.1311	0.1333	0.0420	70.19
Low (k=3)	2,542.4	6,174.4	49,940.0	16.4	0.1751	0.1817	0.0467	63.98

(b) Manpower and equipment requirements

Repair Activity	Manpower Requirement h_{jg} (Man-days/Production Day)				Equipment Requirement q_{jr} (Equipment-days/Production Day)					
	g = 1	g = 2	g = 3	g = 4	r = 1	r = 2	r = 3	r = 4	r = 5	r = 6
j = 1	0	2	4	0	1	0	1	0	0	0
j = 2	1	1	5	1	1	1	0	0	0	1
j = 3	1	3	5	2	3	1	1	1	0	1
j = 4	1	2	2	4	2	1	0	1	0	0

Note: Manpower types 1 to 4 represent supervisors, drivers, laborers, and equipment operators, respectively; equipment types 1 to 6 represent dump trucks, pickup trucks, crew cabs, distributors, loaders, and rollers, respectively.

(c) Pavement Repair Priority Weighting Factors, F_{ijk}

Highway Class	Need-Urgency Level k	Shallow Patching j = 1	Deep Patching j = 2	Premix Leveling j = 3	Crack Sealing j = 4
Urban Interstate (g = 1)	k = 1 (High)	90	100	70	50
	k = 2 (Medium)	63	90	63	45
	k = 3 (Low)	54	60	42	30
Urban Arterial (g = 2)	k = 1 (High)	72	80	56	40
	k = 2 (Medium)	54	70	49	35
	k = 3 (Low)	45	50	35	25
Rural Interstate (g = 3)	k = 1 (High)	76.5	85	59.5	42.5
	k = 2 (Medium)	58.5	75	52.5	37.5
	k = 3 (Low)	40.5	45	31.5	22.5
Rural Primary (g = 4)	k = 1 (High)	70.5	65	45.5	32.5
	k = 2 (Medium)	36	40	28	20
	k = 3 (Low)	18	20	14	10

Generation of Offspring

The crossover operation and mutation operation are the two most commonly used GA operations for generating offspring from parent solutions. Figure 4 illustrates the mechanism of a simple crossover operation and that of a simple mutation operation. Figure 4(*a*) shows a simple crossover operation with a single cross point on two genotypes. A common point is randomly chosen, and the two parts of each genotype are swapped to create two offspring. Each offspring therefore consists of a part of each parent. In the case of a simple mutation operation on the binary genotype shown in Figure 4(*b*), a random number is generated for each allele and an allele is mutated if the random number generated is less than a predetermined number.

TABLE 3 Repair Needs Data and Constraint Information

(a) Repair work requirements in workdays, T_{ijk} / U_{ijk}

Highway Class	Need-Urgency Level k	Shallow Patching j = 1	Deep Patching j = 2	Premix Leveling j = 3	Crack Sealing j = 4
Urban Interstate (g = 1)	k = 1 (High)	4	6	8	2
	k = 2 (Medium)	6	4	2	3
	k = 3 (Low)	3	25	13	18
Urban Arterial (g = 2)	k = 1 (High)	2	6	9	2
	k = 2 (Medium)	2	10	8	8
	k = 3 (Low)	4	20	15	15
Rural Interstate (g = 3)	k = 1 (High)	5	8	6	5
	k = 2 (Medium)	5	2	10	10
	k = 3 (Low)	5	15	15	10
Rural Primary (g = 4)	k = 1 (High)	3	4	8	4
	k = 2 (Medium)	4	16	12	20
	k = 3 (Low)	15	15	18	15

(b) Rehabilitation constraint factors, γ_{ijk}

Highway Class	Need-Urgency Level k	Shallow Patching j = 1	Deep Patching j = 2	Premix Leveling j = 3	Crack Sealing j = 4
Urban Interstate (g = 1)	k = 1 (High)	0.82	0.83	1.00	0.80
	k = 2 (Medium)	0.70	0.90	0.90	1.00
	k = 3 (Low)	1.00	1.00	1.00	1.00
Urban Arterial (g = 2)	k = 1 (High)	0.93	1.00	1.00	0.92
	k = 2 (Medium)	0.84	1.00	1.00	0.96
	k = 3 (Low)	0.81	1.00	1.00	0.90
Rural Interstate (g = 3)	k = 1 (High)	0.92	1.00	1.00	0.83
	k = 2 (Medium)	0.78	1.00	1.00	0.91
	k = 3 (Low)	0.80	1.00	1.00	0.96
Rural Primary (g = 4)	k = 1 (High)	1.00	1.00	1.00	1.00
	k = 2 (Medium)	1.00	1.00	1.00	1.00
	k = 3 (Low)	1.00	1.00	1.00	1.00

(c) Resource constraints and other input information

Parameter	Value
Analysis Period	D = 45 working days
Budget Allocation	b_1 = \$18,000 b_2 = \$20,000 b_3 = 13,000 b_4 = 9,000
Manpower Availability	H_1 = 90 mandays H_2 = 135 mandays H_3 = 270 mandays H_4 = 90 mandays
Equipment Availability	Q_1 = 135 equipment days Q_2 = 45 equipment days Q_3 = 45 equipment days Q_4 = 45 equipment days Q_5 = 45 equipment days Q_6 = 45 equipment days

In the present example problem the simple crossover and mutation operations were found to be ineffective because they consistently led to more than 80 percent invalid offspring. This was because the problem was highly constrained. Various approaches have been used to handle constrained problems in GAs. One approach is to apply a penalty weightage to the objective function value of invalid offspring (*11*). Other approaches use what are known as *decoder* or *repair algorithms* to avoid creating invalid offspring (*14,15*).

The present study adopted an approach that made use of specialized GA operations to handle the constraints. These specialized operations were the single arithmetic crossover and

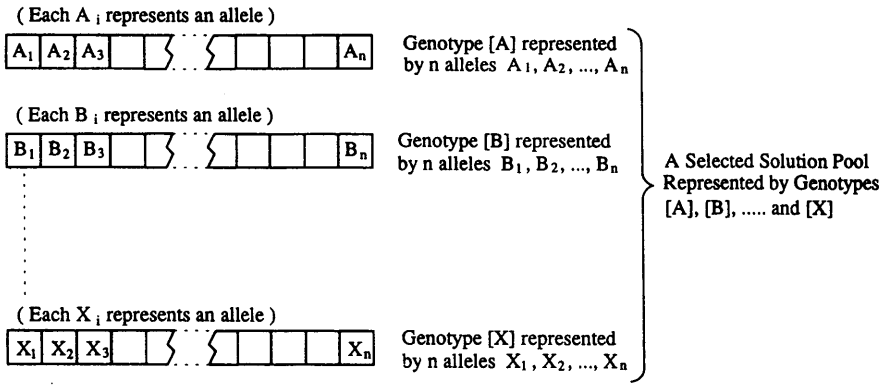

FIGURE 2 GA representation of knowledge.

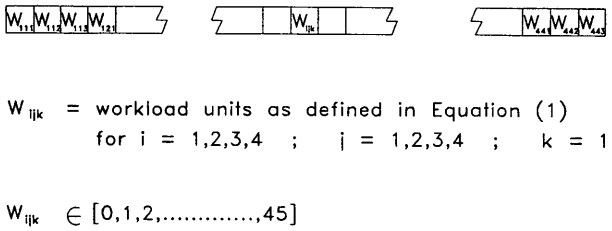

FIGURE 3 Integer coding for genetic representation of example problem.

the nonuniform mutation (*14*). The relative probabilities of applying the crossover and mutation operations were selected as 0.8 and 0.2, respectively. These two operations function as follows. When a single arithmetic crossover is operated on two parent genotypes $<X_1, X_2, \ldots, X_n>$ and $<Y_1, Y_2, \ldots, Y_n>$, the resulting offspring are $<X_1, X_2, \ldots, X_k', \ldots, X_n>$ and $<Y_1, Y_2, \ldots, Y_k', \ldots, Y_n>$, where $k \in [1, n]$, $X_k' = q\,Y_k + (1 - q)\,X_k$, $Y_k' = q\,X_k + (1 - q)\,Y_k$, and q is a random decimal value between 0 and 1.

When a nonuniform mutation operation is executed on a parent genotype $<X_1, X_2, \ldots, X_n>$ and X_k is the gene to be mutated, the resulting offspring would be $<X_1, X_2, \ldots, X_k', \ldots, X_n>$, where X_k' has one of the following values: $(0.2\,q\,X_{max})$ when $X_k = X_{max}$, $(1 - 0.2q)\,X_{max}$ when $X_k = 0$, or $(X_k + pqz)$ when $0 < X_k < X_{max}$. X_{max} is the maximum permissible value of X_k, q is a random decimal value between 0 and 1, p is either $+1$ or -1 with equal probability decided randomly by the program, and z is minimum $(X_k, X_{max} - X_k)$.

Formation of New Parent Pool

In the offspring generation phase 160 offspring were generated from the 80 solutions in the parent pool. The next step was to select 80 solutions from the offspring pool to form the next parent pool. Each offspring was first evaluated by means of the objective function to arrive at the so-called fitness value of the offspring. The top 80 genotypes in terms of fitness were then selected to form the new parent pool. This process ensures that the GA search is always directed toward solutions that return better values of fitness (i.e., the values of objective function) as the solution process proceeds.

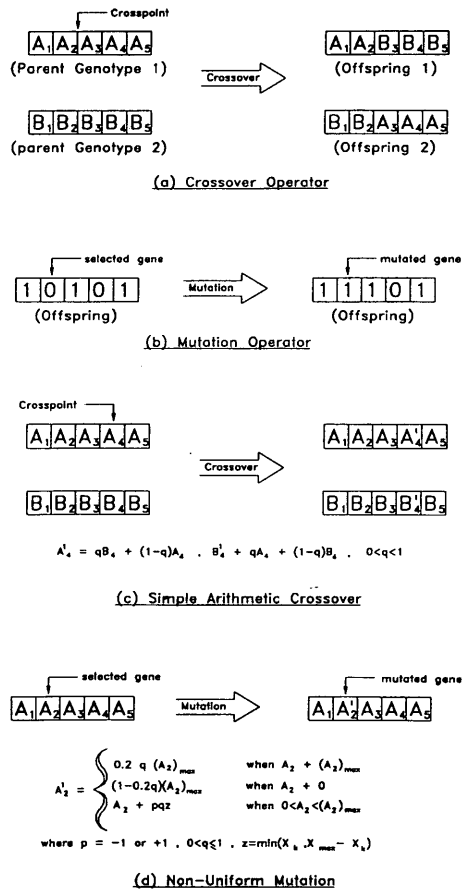

FIGURE 4 Examples of genetic operations.

Performance Evaluation of Solutions

There are four possible measures for assessing the performance of GA solutions. The first measure, known as the *online performance,* is the running average of the value of the objective function of all valid genotypes that have been generated. The second measure, known as the *offline parent-pool performance,* records the average of the objective function values of the genotypes of each parent pool selected. The third measure, known as the *offline offspring-pool performance,* records the average of the objective function values of the genotypes of each pool of offsprings generated. Finally, the *best solution* is the value of the objective function of the best genotype that has been generated. An example is given in Figure 5, which plots the four performance measures against the number of generations on the basis of the results of one of the GA solution runs obtained in the study. The best solution criterion is typically used to compare the performance of GAs, whereas the online performance and the offline performance are often used to monitor the convergence of GA solutions.

Convergence of GA Solutions

The typical trend of convergence of GA solutions is clearly displayed in Figure 5. When the curves of best solution and offline parent-pool performance are considered, it can seen that the GA solutions converged after the 28th generation, although the best solution was achieved at the 23rd generation. The average of parent-pool solutions lagged the best solution in terms of convergence, as expected. The fluctuations of the offline offspring-pool performance curve, after convergence had been achieved by the best-solution and the offline parent-pool performance curves, indicate the on-going GA mechanism of search for possible improvements. In comparison with these three performance measures, the online performance does not appear to provide as good an indication of the trend of convergence.

Figure 6 shows the best solution performance curves for four solution runs of the GA program for the example problem. The number of generations at which the best solution was reached varied from the 14th generation in Solution Run 1 to the 26th generation in Solution Run 3. The offline parent-pool performance curves in Figure 7(*a*) indicate that convergence of the parent-pool solutions occurred at the 19th generation for Solution Run 1 and at the 30th generation for Solution Runs 3 and 4. Figure 7(*b*) plots the offline offspring-pool performance for the four GA runs. All four curves exhibit the postconvergence fluctuations typical of the response of offspring-pool solutions described in the preceding paragraph.

COMPARISON OF INTEGER PROGRAMMING AND GAs

GA Solutions Versus Integer-Programming Solution

Table 4 presents the GA solutions obtained in the study together with the integer-programming solution produced by Fwa et al. (*3*). Table 4(*a*) shows that all the four GA runs could produce good solutions with objective function values comparable to those achieved by the integer programming solution. The improvements over the integer programming solution were 0.57, 0.21, 4.93, and 3.95 percent for GA Solution Runs 1, 2, 3, and 4, respectively. Table 4(*b*) shows the output values of decision variables W_{ijk} for all five solutions. For easy comparison Table 5 summarizes the results by highway class and pavement repair activity type in terms of the decision variables W_{ijk}. These answers in W_{ijk} can be converted into workload measurement units by multiplying the production rates given in Table 2(*a*).

It is apparent from the results in Tables 4 and 5 that the choices of pavement repair activities were different among the five solutions, although the concentration of activities in all of the solutions

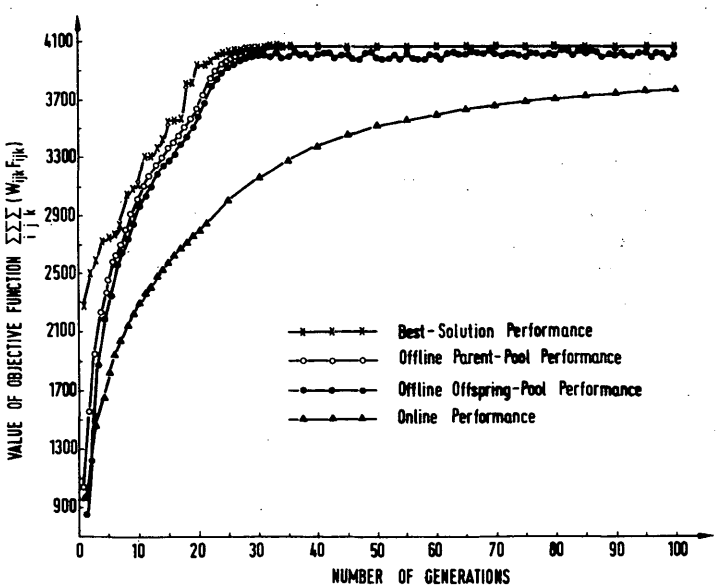

FIGURE 5 Performance evaluation of GA solutions.

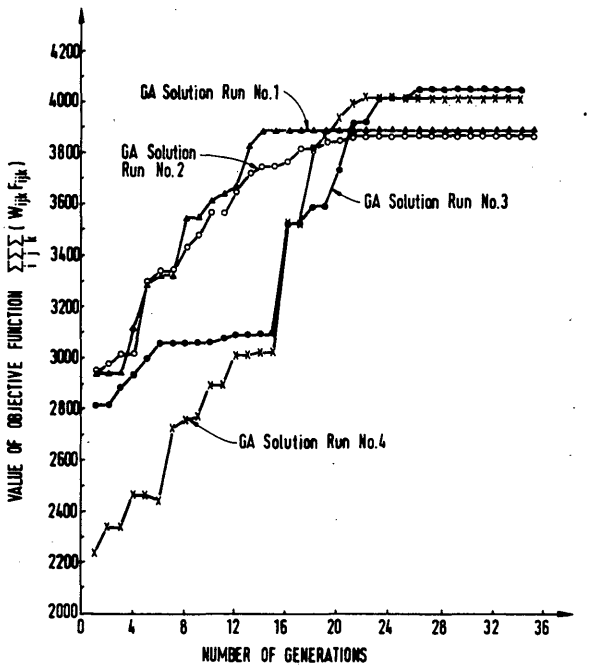

FIGURE 6 Best-solution performance curves for four GA solution runs.

was heavily influenced by the priority weighting factors [see Table 2(*c*)]. This is clearly depicted in Figure 8, in which the results of Table 5 are presented graphically. In general, as shown in Table 2(*c*), urban Interstate ($i = 1$) received the highest priority, whereas rural primary received the lowest; shallow and deep patching ($j = 1$ and $j = 2$, respectively) also had higher priorities than the other two repair activities. These priority patterns are generally reflected in all five solutions. It should be noted; however, that the need to maximize the objective function had the ultimate impact on the solutions. For example, on the whole more workdays of shallow patching ($j = 1$) than deep patching ($j = 2$) were assigned, even though the latter had higher priority.

Comments on GA Application to Pavement Management Problems

GAs have the inherent property of being able to process a large number of similar string structures. Holland (*6*) gave the name of *intrinsic parallelism* to this property, which is related to the notion of schema. A schema can be defined as a pattern-matching device or a similarity template that describes a subset of strings with similarities at certain string positions. For example, a binary code schema $111xx$ describes a subset of four strings (11100, 11101, 11110, 11101). In each generation GA operations process approximately a total of n^3 schemata (*6*), where n is the size of the parent pool. This property gives GAs great computational leverage.

With the stochastic generation of offspring in their search for new solutions, GAs represent a global search process. However, they are not simply random procedures because the offspring are generated from a parent pool of solutions that have been selected on the basis of their fitness. GAs are therefore able to efficiently exploit past information to explore new regions of the decision space with a high probability of finding solutions with improved fitness.

The choice of genetic operators in offspring generation is an important phase in GAs that can be used to provide meaningful ways of combining genetic information from different genotypes in the parent pool. Care should also be taken that the selected genetic operations do not lead to the creation of excessive numbers of invalid offspring or contribute to premature convergence. For example, the crossover operation alone would not be sufficient because *lost alleles* cannot be recovered, hence leading to premature convergence (*16*). A lost allele occurs when the entire parent pool or all offspring solutions have the same value for a particular gene. This problem is overcome by the use of the mutation operation that helps to maintain the genetic diversity and keep the gene pool well-stocked. It is significant that the mutation operation allows the search to reach new areas of the parameter space.

It is observed from the example application (e.g. Table 4 and Figures 6 and 7) that there were differences among the solutions obtained from different GA runs because of the stochastic nature of the technique. This is a genetic phenomenon known as *genetic drift* (*11*). Genetic drift is common and expected in applications to combinatorial optimization problems, and the differences are usually small. It should be noted that it is not possible to perform an ex-

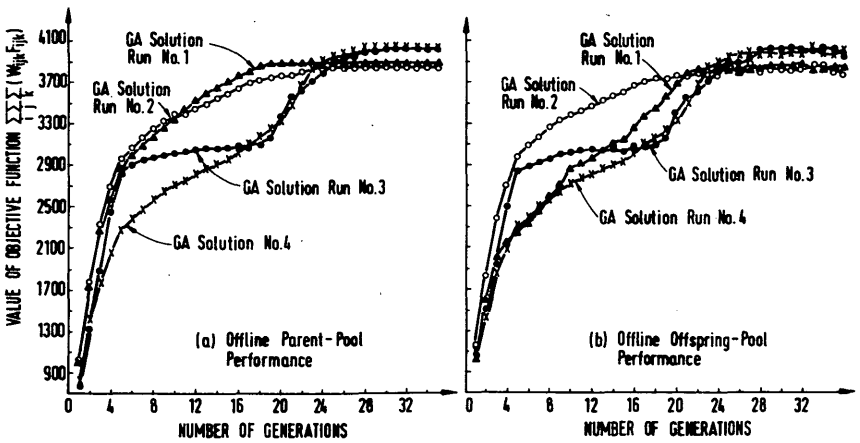

FIGURE 7 Performance curves for offline parent-pool solutions and offline offspring-pool solutions.

TABLE 4 Solutions of Example Problem by Integer Programming and GAs

(a) Values of Objective Function of Solutions

Method of Analysis	Integer Programming	GA Solution Run No. 1	GA Solution Run No. 2	GA Solution Run No. 3	GA Solution Run No. 4
Value of Objective Function	3865.5	3887.5	3873.6	4056.0	4018.0
Percent Improvement	--	0.57%	0.21%	4.93%	3.95%

(b) Values of Decision Variables W_{ijk}

W_{ijk}	IP	GA1	GA2	GA3	GA4	W_{ijk}	IP	GA1	GA2	GA3	GA4
W111	3	2	2	2	3	W311	4	3	4	4	4
W112	4	3	4	4	3	W312	3	2	3	3	3
W113	3	2	3	3	3	W313	4	3	4	4	4
W121	4	4	4	4	4	W321	0	0	3	0	0
W122	3	2	2	3	3	W322	2	2	2	2	2
W123	0	0	0	0	0	W323	0	0	0	0	0
W131	0	1	0	6	2	W331	0	5	5	0	5
W132	1	1	1	0	0	W332	0	0	0	0	0
W133	4	1	3	4	4	W333	0	0	0	0	0
W141	1	1	1	0	0	W341	4	3	3	4	4
W142	3	2	3	3	3	W342	1	1	1	0	3
W143	0	0	0	0	0	W343	0	0	0	0	0
W211	1	1	1	0	0	W411	3	2	2	3	3
W212	1	1	1	0	0	W412	4	3	3	4	4
W213	3	2	3	3	3	W413	1	1	1	0	0
W221	0	0	0	0	0	W421	0	0	0	0	0
W222	6	5	3	6	6	W422	0	0	0	0	0
W223	0	2	0	0	0	W423	0	0	0	0	0
W231	0	7	0	0	0	W431	0	0	0	0	0
W232	0	0	0	0	0	W432	0	0	0	0	0
W233	0	0	0	0	0	W433	0	0	0	0	0
W241	1	1	1	1	0	W441	0	0	0	2	0
W242	0	2	1	0	0	W422	0	0	0	0	0
W243	0	0	0	0	0	W423	0	0	0	0	0

Note: IP = integer-programming solution
GA1 = genetic-algorithm solution No. 1
GA2 = genetic-algorithm solution No. 2
GA3 = genetic-algorithm solution No. 3
GA4 = genetic-algorithm solution No. 4

TABLE 5 Summary of Workloads (W_{ijk}) by Highway Class and Repair Activity for Different Solutions

Highway Class	Shallow Patching (j = 1)	Deep Patching (j = 2)	Premix Leveling (j = 3)	Crack Sealing (j = 4)
i = 1	(10, 7, 9, 10, 9)	(7, 6, 6, 7, 9)	(5, 3, 4, 10, 9)	(4, 3, 4, 3, 3)
i = 2	(5, 4, 4, 3, 3)	(6, 5, 3, 6, 6)	(0, 7, 0, 0, 0)	(1, 3, 2, 1, 0)
i = 3	(11, 8, 11, 11, 11)	(2, 2, 5, 2, 2)	(0, 5, 5, 0, 5)	(5, 4, 4, 4, 7)
i = 4	(8, 6, 6, 4, 7)	(0, 0, 0, 0, 0)	(0, 0, 0, 0, 0)	(0, 0, 0, 0, 0)

Note: (1) Highway class i = 1, 2, 3 and 4 represent urban interstate, urban arterial, rural interstate, and rural primary respectively.
(2) The set of 5 numbers in parentheses represents five solutions in the following order: integer programming, GA run 1, GA run 2, GA run 3 and GA run 4.

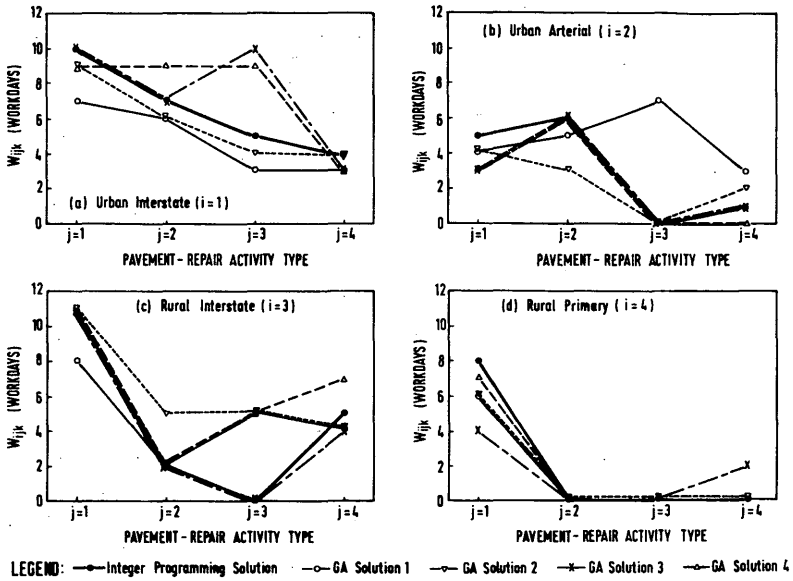

FIGURE 8 Comparison of solutions by integer programming and GAs.

haustive search for such problems, and what is important is the ability to reliably obtain a good and acceptable solution within a practical time frame. For the example problem, each GA solution run took less than 6 hr on a PC386 executing at a clock speed of 33 MHz.

Traditional methods of optimization most often are structurally rigid, with system models and improvement algorithms usually fixed in form. GAs do not have these major shortcomings because they require only payoff information defined by the objective function. This presents yet another attractive aspect of GAs in that it is relatively easy to modify the objective function to suit the user's requirements without affecting the efficiency of the GA search. For example, in pavement management at the network level the objective function could be to maximize production, as was specified in the example problem, to minimize the present worth of pavement expenditures over the analysis period, to maximize the use of allocated budgets, or to minimize the fluctuations of yearly demand for pavement expenditures.

CONCLUSIONS

This paper demonstrates the application of GAs to pavement management problems by solving a network-level pavement repair programming problem. The combinatorial explosion problem associated with a typical network-level pavement management programming analysis makes GAs an attractive technique for highway engineers and planners. The global search ability and flexibility and ease with which GAs can handle different objective functions facilitate comparison of the relative impacts of different strategies.

Genetic representation and the choice of GA operators are two major elements in the GA formulation of the problem analyzed. The considerations involved in the selection of both were illustrated in the paper through the example application of GA to a network-level pavement programming problem. In comparison with the solution obtained by an integer programming method, the example application showed that consistently good solutions can be achieved by GAs within a practical computation time on a personal computer.

REFERENCES

1. Feighan, K. J., M. Y. Shahin, and K. C. Sinha. A Dynamic Programming Approach to Optimization for Pavement Management Systems. *Proc., 2nd North American Conference on Managing Pavements,* Vol. 2, Nov. 1987, Toronto Ontario, Canada, pp. 2.195–2.206.
2. Markow, M. J., B. D. Brademeyer, J. Sherwood, and W.J. Kenis. The Economic Optimization of Pavement Maintenance and Rehabilitation Policy. *Proc., 2nd North American Conference on Managing Pavements,* Vol. 2, Nov. 1987, Toronto, Ontario, Canada, pp. 2.169–2.182.
3. Fwa, T. F., K. C. Sinha, and J. D. N. Riverson. Highway Routine Maintenance Programming at Network Level. *Journal of Transportation Engineering,* Vol. 114, No. 5, 1988, pp. 539–554.
4. Lytton, R. L. From Ranking to True Optimization. *Proc., North American Pavement Management Conference,* Vol. 3, March 18–21., 1985, Toronto, Ontario, Canada, pp. 5.3–5.18.
5. *Pavement Management Systems.* Organization for Economic Cooperation and Development, Paris, 1987.
6. Holland, J. H. *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, 1975.
7. Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Company, Inc., Reading, Mass., 1989.
8. Bethke, A. D. *Genetic Algorithms as Function Optimizers.* Technical Report 212. Logic of Computers Group, University of Michigan, Ann Arbor, 1981.
9. Englander, A. C. Machine Learning of Visual Recognition Using Genetic Algorithms. *Proc., International Conference on Genetic Algorithms,* (J.J. Grefenstette, ed.). Carnegie-Mellon University, Pittsburgh, Pa., pp. 197–202.
10. Staduyk I. Schema Recombination in Pattern Recognition Problem. Genetic Algorithms and Their Applications. *Proc., 2nd International Conference on Genetic Algorithms.* Morgan Kaufmann Publishers, Los Altos, Calif., 1991.

11. Goldberg, D. E. *Computer-Aided Pipeline Operation Using Genetic Algorithms and Rule Learning.* Ph.D. dissertation. University of Michigan, Ann Arbor, 1983.
12. Khoogar, A. R. Genetic Algorithm Solutions for Inverse Robot Kinematics. *Proc., ACM Student Conference.* University of Alabama, Birmingham, 1978.
13. Cohen, C., and J. Stein. *Multi-Purpose Optimization Scheme—User's Guide,* Version 4, Manual 320. Vogelback Computer Centre, Northwestern University, Evanston, Ill., 1978.
14. Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer-Verlag, Berlin, Germany, 1992.
15. Davis, L. *Genetic Algorithms and Simulated Annealing.* Morgan Kauffmann Publishers, Inc., Los Angeles, Calif., 1987.
16. Baker, J. E. Adaptive Selection Methods for Genetic Algorithms. *Proc., International Conference on Genetic Algorithms* (J.J. Grefenstette, ed.). Carnegie-Mellon University, Pittsburgh, Pa., 1985, pp. 101–111.