Genetic Algorithm: A Tool for Optimization

Ripunjoy Gogoi
Assistant Professor, Civil Engineering
Amity University Madhya Pradesh, Gwalior, India

Abstract: The formulation of engineering optimization problem is sometime very complex to solve. Due to complexity of the problems, traditional methods are inefficient to give proper solution. In such a scenario, non traditional method such as Genetic algorithm is able to give proper solution. In this paper the working principle of the standard genetic algorithm has been discussed with its advantages.

***IndexTerms* - *binary GA, optimization*.**

I. Introduction

Genetic algorithm (GA) works on the principle of natural genetics and natural selection and it is an iterative process, making improvements in each successive iteration, called generations [1,2]. GA emphasis the previously found good results in its next generation by operators called as: crossover, mutation and reproduction. In each generation, a set of population is evaluated called solutions. Each solution in GA is represented by a string. The paper first discusses the typical formulation of an engineering optimization problem. Then the standard working principle of GA is discussed. At the end conclusion is presented.

II. TYPICAL FORM OF OPTIMIZATION PROBLEM

A typical engineering optimization problem consist of: (a) an objective function and (b) constraints called decision variables. Figure 2 shows the various components of engineering optimization problem. The objective function is either maximization type or minimization type. The problem of optimization is shown with respect to highway problem. The optimization problem is multi objective function and the objective functions are; (a) minimization of highway agencies cost (b) minimization of cost of vehicle users' cost etc. For the problem to be solved, the information required is shown in the category of data and model. The constraints or the decision variables are: (a) total budget should not be crossed, (b) not exceeding the number of maintenance activities that are possible etc.

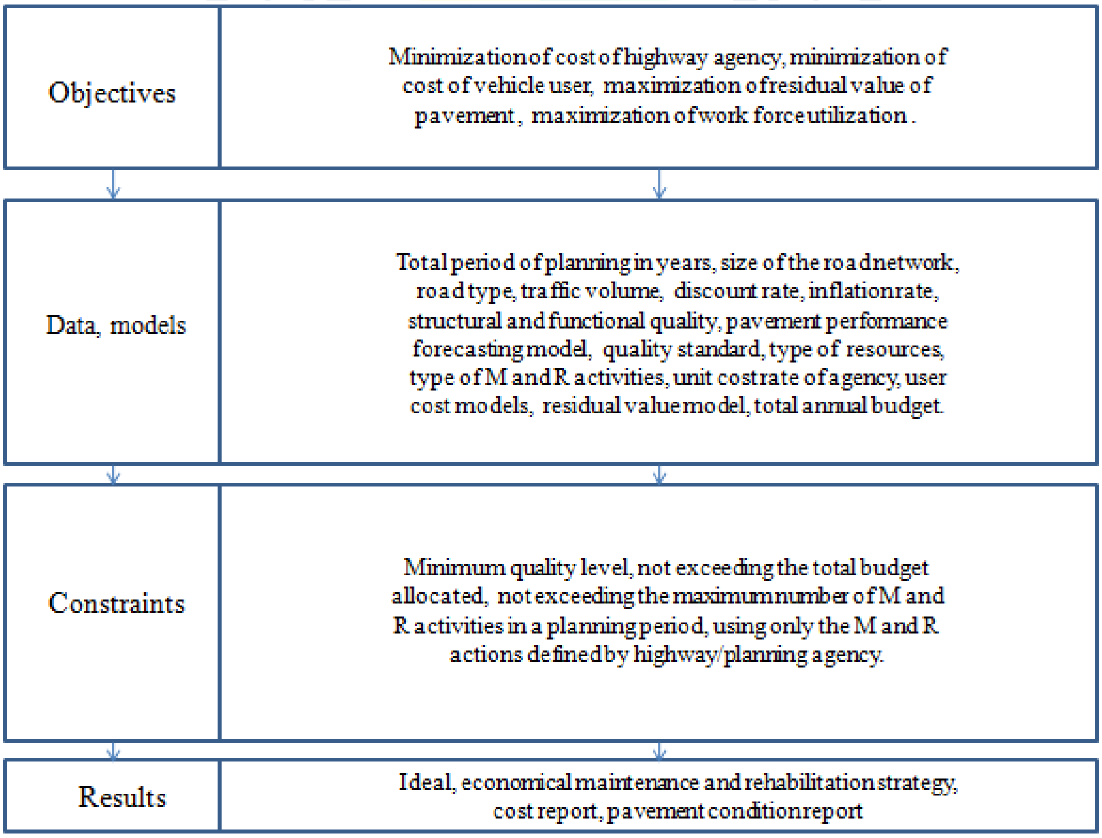

Figure 2: Schematic showing the components of engineering optimization problem

The typical form of an optimization problem is shown below:

Objective function:

Minimization or maximization:

$f(x)$

subject to:

$g(x) < g(x)_{max}$

$h(x) < h(x)_{max}$

$p(x)_{min} < p(x)$

III. WORKING PRINCIPLE OF STANDARD GA

In binary GA, decision variables are represented by a string (chromosome) of binary alphabets (genes). The size of the string is decided based on the total number of values to be represented for a decision variable. For example, if there are four values of a decision variable, which are 100, 200, 300 and 400, a two bit binary coding (consist of '0' and '1') can be used to represent the values. The representation would be as follows: (00) is 100, (01) is 200, (10) is 300 and (11) is 400. Similarly, values of other decision variables can be represented by changing the bit size of the binary codes as per the total number of values present. Figure 2 shows the basic structure of the binary GA based procedure to solve the present optimization problem.

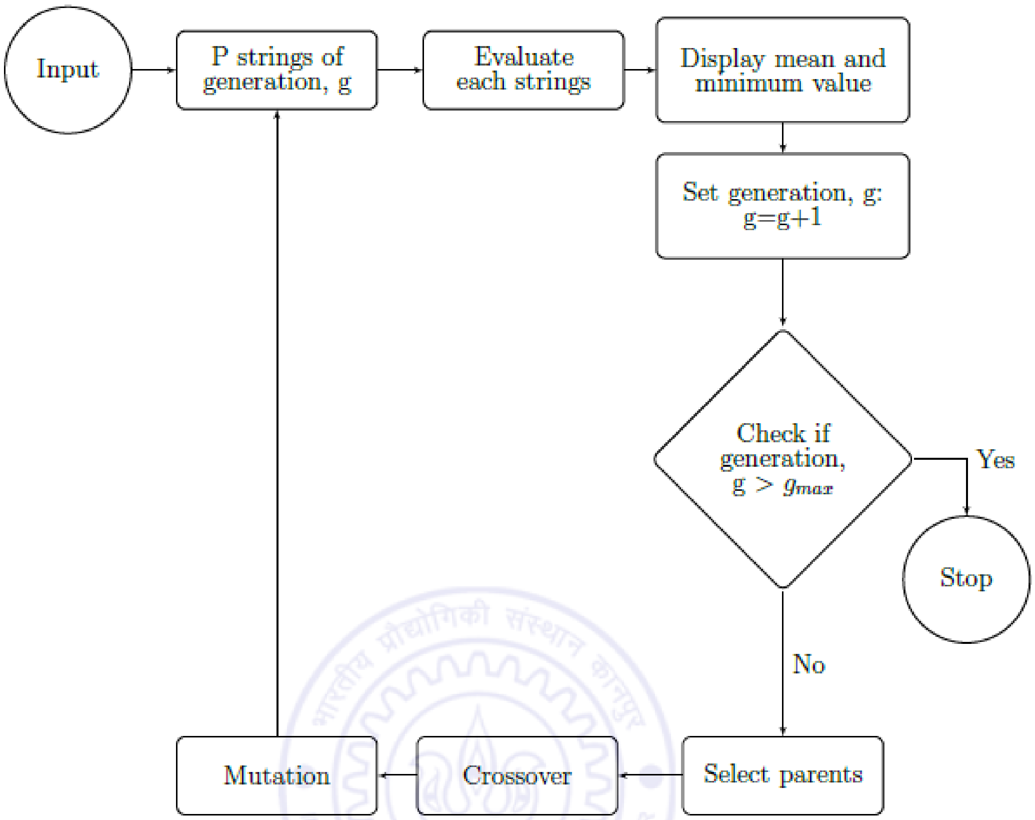

Figure 2: Schematic showing the standard algorithm of GA

At first, for generation g=1, a set of population (say P) representing the values of decision variables is provided as input. These initial set of population are created randomly. This population includes multiple individuals and is represented by binary bit codes. A single individual of a population represented with binary codes are called 'strings' comprises of all the four decision variables represented with binary codes and are called 'substrings'. At generation g=1, each string is first decoded (converted to real values of decision variables) and then evaluated one by one to calculate the objective function value (to know the minimum total cost), which is a measure of the 'goodness' of the string. In GA, the value of objective function is termed as 'fitness'. The minimum total cost and the mean value of total costs (for all the strings) for that particular generation is stored. For the next generation (g=g+1), GA operators: reproduction, crossover and mutation are used to create a new population. Reproduction or selection operator helps in making copies of better strings in a population. Here 'tournament' selection scheme is used to select the best strings from the previous generation. This scheme involves running several 'tournaments' between strings, chosen randomly from the population. Winner of each tournament (with best 'fitness') is selected for crossover. For example, in this problem the tournament size is taken to be two, which means a tournament will be played between two strings and the best fit value will be chosen for crossover. Reproduction only makes copies of good strings but it does not create any new strings. Therefore, to create a new string, a crossover operation is carried out. A crossover operator recombines two strings to create a better string, expecting that it will lead to a better fitness. A single point crossover scheme is used for the current optimization problem. In this crossover, two random strings are chosen (as an example two seven bit string is used to explain the crossover scheme below). A crossover point is selected randomly (represented by the vertical line). At this point, the tails of the strings (can be called parent strings) are swapped to create a new strings called the children (off springs). The main motive of this operation is to create strings which are better than the parent strings. After the crossover stage is over, operator mutation is used to create diversity in the new population. Under this action, bits of a string are flipped randomly with a very low probability rate. For example, in a binary bit (consisting 0's and 1's), it changes a 1 to 0 and vice versa. In this problem, the mutation probability is assumed as 0.01 and thus, for a population size of 30, with each string length of 16, the total number bits that will alter is equal to five ($30 \times 16 \times 0.01$) and rounding off to nearest highest integer value). Once, the new strings are created, they are evaluated by decoding and the total cost is calculated. Again, the minimum value of the total cost and the mean value of the total costs for that particular generation will be stored. This process will continue, until in the subsequent generations, the minimum total cost value stabilizes. It is expected that at generation 'g=g_{max}' the cost will stabilize and hence 'g_{max}' is set as the maximum allowable generation for the problem.

IV. CONCLUSIONS

Genetic algorithm is a very efficient tool to solve engineering optimization problem. It can be employed to solve optimization problems. When traditional optimization tools are unable to solve optimization problems, genetic algorithm can be used.

REFERENCES

[1] Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning, 1989.

[2] Deb, K. (2012). Optimization for engineering design: Algorithms and Examples. PHI Learning.