International Journal of Research in Engineering and Applied Sciences(IJREAS)
Available online at http://euroasiapub.org/journals.php
Vol. 7 Issue 9, September-2017, pp. 48~58
ISSN (O): 2249-3905, ISSN(P): 2349-6525 | Impact Factor: 7.196

GENETIC ALGORITHM: APPLICATIONS TO LINEAR AND INTEGER PROGRAMMING PROBLEMS

Sangeeta Singh[1],
Amity School of Applied Sciences,
Amity University Lucknow Campus, India

Richa Mehrotra[2],
Amity School of Applied Sciences,
Amity University Lucknow Campus, India

Abstract

Linear programming problem has powerful capabilities that enable businesses to reduce costs, improve profitability, use resources effectively, reduce risks and provide benefits in many other key dimensions. Integer programming problem is a special case of linear programming problem because this optimization technique provides optimal integer solution of the programming problem. This technique plays important role in business and industry problems. In this paper, we have discussed linear and integer programming problems and their various applications. Apart from regular methods of solving these problems, we have studied a heuristic search approach Genetic Algorithm in optimization.

Keywords: Linear programming problems, integer programming problems, methods for optimizing LPP and IPP, Genetic algorithm, applications of LPP and IPP, genetic algorithm for solving linear and integer programming problems.

Introduction

Linear programming is one of the most important optimization techniques which are developed in the field of operation research. It is a method or mathematical technique to find the best outcome (such as maximum profit or lowest cost) of linear function. A linear programming problem, maximizing or minimizing a linear function, can be of several decision variables subjected to linear constraints where constraints can be either inequalities or equalities. A linear programming problem consists of three components:

- Decision variable
- Objective function
- Constraints

General form: Suppose x_1, x_2, --------, x_n are n variables. Find the values of n decision variables which maximize or minimize the objective function $z = c_1x_1 + c_2x_2 + \text{------} + c_nx_n$ where all cj are constants. j = 1, 2, ---- n. and satisfy m constraints

$$a_{11}x_1 + a_{12}x_2 + \text{----------} + a_{1j}x_j + \text{-----} + a_{1n}x_n \ (\leq \text{ or } \geq \text{ or } =)\ b_1$$
$$a_{21}x_1 + a_{22}x_2 + \text{----------} + a_{2j}x_j + \text{------} + a_{2n}x_n \ (\leq \text{ or } \geq \text{ or } =)\ b_2$$

.
.

$a_{m1}x_1 + a_{m2}x_2 +$ ---------- $+ a_{mj}x_j +$ ------ $+ a_{mn}x_n$ ($\leq$ or $\geq$ or $=$) b_m

where constraints are in the form of equalities or inequalities and decision variables satisfy non – negativity constraints $x_1 \geq 0$, $x_2 \geq 0$, --------, $x_j \geq 0$,------, $x_n \geq 0$. Here values of right side constants bi (i = 1, 2, 3, ----- m) will be non-negative. If any bi has negative values then it can be changed into positive value by multiplying (- 1) on the both sides of corresponding constraint.

Integer Programming Problem- A Special Case of Linear Programming Problem

As the name, itself indicates, integer programming problem is a special case of linear programming problem in which all or some decision variables are restricted to non- negative integer values.

Types of Integer Programming Problem

- Pure integer programming problem
- Mixed integer programming problem
- Zero one integer programming problem

The main aim of integer programming is to find optimal integer solution of the problem. We can determine the optimal integer solution of the integer programming problem by using regular simplex and by rounding off the fractional values occurring in the optimal solution. But in this case it is not necessary that the deviation of our solution from the exact optimal solution be very less which is negligible. That is, it may be large. We cannot find feasible solution of the integer programming problem by simplex method.

Integer linear programming evolved with the implementation of linear programming problem itself when scientists realized there were a need for certain models, especially models, developed in business problems, to have an integer-only constraint.

Literature survey

The first linear programming formulation of a problem was given by Leonid Kantorovich in 1939 who was a Russian economist, who also proposed a method for optimizing it. In 1941, Frank Lauren Hitchcock also formulated transportation problem as a linear programming problem and gave a solution which is very similar to the simplex algorithm.

During 1946-1947, George B. Dantzig developed linear programming formulation to use for planning problems in U.S Air force. In 1947 Dantzig arrange a meeting with John Von Neumann to discuss his simplex method.

The linear programming problem was first shown to be solvable in polynomial time by Leonid Khachiyan in 1979, but a larger theoretical and practical breakthrough in the field came in 1984 when Narendra Karmarker introduced a new interior point method for solving linear programming problem.

A major contributor to the evolution of integer programming was George Dantzig (1951). His work, the travelling salesman problem, was the original branch-and-cut algorithm (1954) that forms the basis of integer systems that are used to solve models in areas such as transportation, telecommunication, manufacturing, supply chain.

In 1956 R.E. Gomory suggested a systematic procedure to obtain an optimum solution of the all integer programming problem and then extended this method for solving more complicated case of mixed integer programming problem. This method gives optimum integer solution of the

problem in a finite number of iteration by using dual simplex method and this algorithm is known as cutting plane method.
A general method for solving all and mixed integer programming problem, developed by A.H. Land and A.G. Doig (1960), is called branch and bound method. Also Egon blas (1965) introduced an enumerative method for solving Zero – one integer programming problem.

Methods for optimizing Linear and Integer programming problems

a. For linear programming problem:
  - Graphical Method: This method gives the optimal solution of only linear programming problem of two decision variables.
  - Simplex Method: It is most powerful technique for solving three and more than two decision variables linear programming problem.

b. For integer programming problem:
  - Branch and Bound Method: It is most general optimization method and this method can be applied for both all and mixed integer programming problem. It gives optimal solution to the entire space solution
  - Gomory's Cutting Plane Method

Genetic Algorithm

Genetic algorithm is a computerized method for optimizing the optimization problem (optimization problem can be restricted by constraints or not) which is based on the natural selection. Professor Holland had given the concept of genetic algorithm in the mid-sixteen and it was published in 1975 and then some other researchers have contributed to the development of this field.
Genetic algorithm is very different and important technique from other traditional methods. Genetic algorithm is always used with a coding of variable because coding of variable provides discrete search space where function may be continuous. This is the advantage of coding of variable. For using genetic algorithm, definition of objective function, genetic representation and genetic operator should be clear.

Definition: Genetic algorithm is heuristic method and optimization technique for solving optimization problem that mimic the process of natural evolution.

Principle of natural selection: Natural selection is the differential endurance and reproduction of individuals due to differences in phenotype.
Natural selection is key mechanism of development, the change in paternal traits of population over time. Charles Darwin popularized the term "Natural Selection" and compared it with artificial selection. "Select the best, discard the rest".

a- Some Terminologies
Chromosome: A set of genes. Chromosome contains the solution in form of genes.
Gene: gene is a part of chromosome and a gene contains a part of solution. It evaluates the solution. Example: 13426 is a chromosome then 1,3,4,2 and 6 are its genes.
Individual: Same as chromosome.

Population: Number of individuals present with same length of chromosome.
Search space: When we solve any problems, we want to find out some such solutions which are best among other solutions. So the space for all possible feasible solution is called search space.
Fitness: Fitness is a value assigned to an individual and it is based on the how far or close an individual is from the solution.
Fitness function: Fitness function is a function which assigns fitness value to the individual.
Selection: Selecting individuals for creating the next generation.
Recombination or crossover: Genes from parents form in some way the whole new chromosome.

Encoding
The process which is used for representing a solution in the form of string is called encoding and encoding carries necessary information. As in chromosome, each gene represents a particular characteristic of the solution.

Encoding method
There are several methods for encoding, but in genetic algorithm the most common used method of encoding is binary coded.
Chromosomes are string of 1 and 2.Each position in the chromosomes represents a particular characteristic of the problem.

b- Operators and parameter
Selection
Selection is the process that determines which solutions to be conserved and allowed to reproduce and which solution deserve to die out.
The main aim of selection is to give importance to the good solution and eliminate the bad solution in a population where population size is constant. "Select the best, discard the rest".
Functions of selection operator

- Analyze the good solution in population.
- Create multiple copies of good solution.
- Bad solutions eliminate from population so therefore multiple copies of good solutions can be placed in the population.

There are several techniques for applying selection operator-
Tournament Selection

- In tournament selection, several tournaments are played among some individuals. Individuals are taken randomly from the population.
- The winner of each tournament will be selected for the next generation.
- Selection pressure will be varied according to the tournament size i.e. if tournament size is small then selection pressure will also be decrease and vice-versa.
- If tournament size is large then weak individuals will have small chances for selecting in next population.

Roulette wheel and proportionate selection: parents are selected according to their fitness value. The better chromosomes have more chances to be selected.

Chromosomes	Fitness value	% of roulette wheel	EC	AC
1	50	26.88	1.61	2
2	6	3.47	0.19	0
3	36	20.81	1.16	1
4	30	17.34	0.97	1
5	36	20.81	1.16	1
6	28	16.81	0.90	1

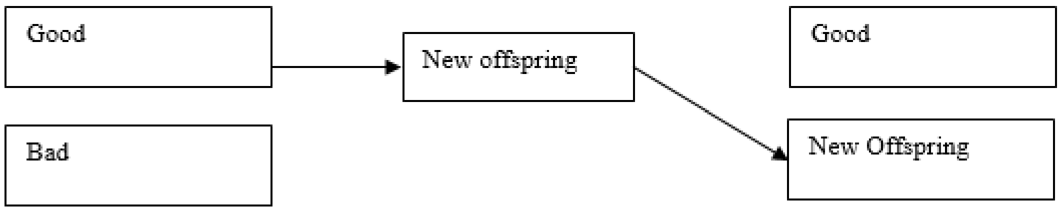
EC = expected count, AC = actual count, %of RW = (Fi /sum of fitness value of all chromosomes) x100, where Fi is the fitness value of ith chromosomes and i = 1, 2...

Rank selection

When value of chromosomes differ very much then more problems occur to form Roulette wheel according to fitness values. For example, if best chromosome has more fitness value and if this chromosome has percentage of RW is 90% then it will occupies 90% circumference in RW and other chromosomes will have less chances to be selected.
Rank selection first ranks the population according to fitness value and then assigns rank to each chromosome according to fitness value. The chromosome which has very much fitness value will have 1st rank and next chromosome will have 2nd rank and so on.

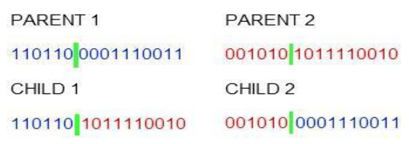
Steady state selection

In steady state selection first the bigger part of population are taken and then few good chromosomes are selected for the creation of new offspring in each process. So some bad chromosomes are eliminated and new offspring is placed in their place and rest of population migrates to next generation.



Crossover

After applying selection operator to create new solutions from existing solutions available in mating pool, crossover operator is used. In crossover operator encoding of solution is necessary. So solutions appear like chromosome. This operator exchange genes between solutions in mating pool.
In crossover, first selection operator selects any two solution string for mating then cross site selects randomly from the mating pool and then position value exchange between two strings.
Example

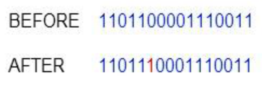


Crossover

Mutation

Mutation is the introduction of new characteristics into the solution string of the population to maintain diversity in the population.

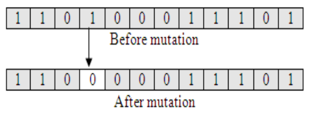
While, crossover has main responsibility to search for optimal solution and mutation is used for this purpose.



Mutation

Binary mutation

Binary mutation operator changes to 1 into 0 and vice versa.



Elitism

When we apply mutation and crossover operator in the population pool then some of the best solution of the population may be destroy.

Where, elitism is conservation of some best solution of the population pool and elitism is defined in percentage or in number.

Literature survey

In operation research, a genetic algorithm is a metaheuristic inspired by the process of natural selection. Genetic algorithm is commonly used to generate high-quality solutions to optimization and search problem by relying on bio-inspired operators such as mutation, crossover and selection.

In 1950, Alan Turing proposed “learning machine’’. This machine was parallel to the principle of evolution. Computer simulation of evolution started as early as in 1954. With the work of nils aall barricelli who was using the computer at institute for advance study in princeton new jersey.

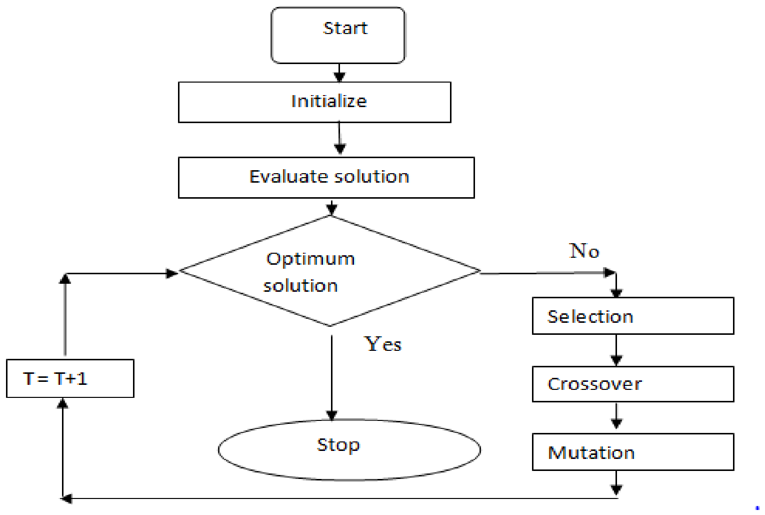
Australian quantative geneticist alex fraser published a series of paper on simulation of artificial selection of organism with multiple laci controlling measurable traits then after all of these, computer simulation of evolution by biologists has been became more common in early 1960s and methods were discussed in books by fraser and burnell (1970) and crosby (1973).

In addition, hans joachim bremermann published a paper in `1960s and he adopted a population of solution to optimization problem, undergoing crossover, mutation, and selection. Also, the elements of modern genetic algorithm were included in research of bremermann. In 1960s – 1970s, rechenberg’s group was able to solve complex engineering problem.

Genetic algorithm in particular became popular through the work of john halland in the early 1970s and particularly his book adaptation in natural and artificial system (1975).

Application of genetic algorithm to LPP

a- Simple genetic algorithm



b- Programming

```
Function sga ( )
{
    Initialize population;
    Calculate fitness function;
    While (fitness value! = termination criteria)
    {
        Selection;
        Crossover;
        Mutation;
        Calculate fitness function;
    }
```

c- Working Rule

Subrata et al. applied Genetic algorithm and Simplex method on the same Linear Programming Problem and find out the difference in the result. The tested linear programming problem is as given below

Maximize F(x1, x2) = 4x1 + 3x2

Subject to constraints $2x1 + 3x2 \le 6$

$-3x1 + 2x2 \le 3$

$2x1 + x2 \le 4$

$0 \le x \le 2$

When we apply simplex method on this linear programming problem then it gives only one set of solution where x1 = 1.5 and x2 = 1 and then F(x1, x2) = 9.

Subrata et al. then proposed a step by step procedure for applying genetic algorithm in a specific optimization problem

Suppose the problem of maximizing the function f(x) = 100x where x is permitted to vary between 0 and 31.

Step 1: We must first code the decision variables of as some finite- length string first to use genetic algorithm. The variable x is coded as a binary unsigned integer of length 5. As with a five-bit (binary digit) unsigned integer we can obtain numbers between 0(00000) and 31 (11111).
Step 2: An initial population is selected at random. The initial population is obtained in the form of binary strings. These strings are converted into their corresponding decimal values to calculate the corresponding fitness function value for each x.

String number	Initial population (Randomly Generated)	x Value (unsigned integer)	f(x) = 100x
1	01101	13	1300
2	11000	24	2400
3	01000	8	800
4	10011	19	1900

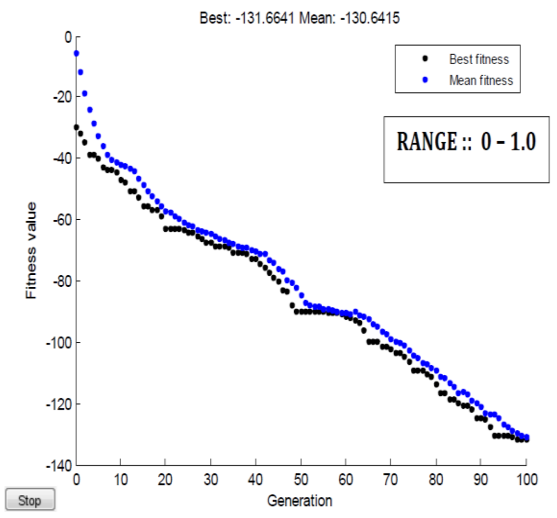
Step 3: Genetic algorithm starts with reproduction of initial population. We choose the mating pool of the next generation by spinning the weighted roulette wheel four times. Actual simulation of this process using coin tosses has resulted in string 1 and string 4 receiving one copy in the mating pool, string 2 receiving two copies, and string 3 receiving no copies. Comparing this with the expected number of copies gives what we should expect, the best get more copies, the average stay even, and the worst die off.
Step 4: With an active pool of strings looking for mates, simple crossover proceeds in two steps:

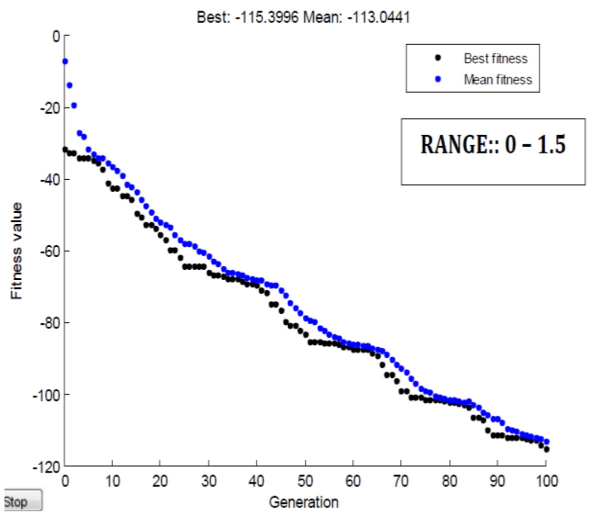
1. Strings are mated randomly
2. Mated string couples cross over

Random selection of mates has selected second string in the mating pool to be mated with the first. With a crossing site of 4, the two strings 01101 and 11000 cross and form two new strings 01100 and 11001. The remaining two strings in the mating pool are crossed at site 2. Here crossover probability is assumed to be unity PC = 1.0.
Step 5: The last operator, mutation is performed on a bit-by-bit basis. Let the probability of mutation in this test is 0.001. With 20 transferred bit positions we should expect 20*0.001 = 0.02 bits to undergo mutation during a given generation. Simulation of this procedure shows that no bits undergo mutation for this probability value. Therefore, no bit positions are changed from 0 to 1 0r 1 to 0 during this generation.

Results and discussion

When we genetic algorithm is applied to the above problem, then a series of solutions is produced and at a certain generation got the greater value of the first chromosome i.e. x1 = 1.6 in comparison with the previous one. It is remarkable benefit of using genetic algorithm. The best fitness, the best individual, best selection graphs determinate with our proposed genetic algorithm in the range 0 – 1.0 and 0-1.5 are given below





Best fitness graph using genetic algorithm (Range 0 – 1.0) Best Fitness Graph Using Genetic Algorithm Range (0 to 1.5)

Feng in 2008 applied genetic algorithm to solve a linear programming problem in which the constraint coefficients were not precisely defined. He used extension principle, interval arithmetic, and α-cut operations for fuzzy computations, and used a penalty method for constraint violations. The proposed approach simulated every fuzzy number by distributing it into certain partition points. GAs were then used to evolve the values in each partition point. As a result, the final values represent the membership grade of that fuzzy number. After calculating the estimated values of each uncertain coefficient, he obtained a defuzzified linear programming problem. The crisp problem can then be solved using GA.

Barboza et al. approached a production scheduling involving diesel storage and distribution in an oil refinery. To find the solution, a Mixed Integer Linear Programming (MILP) was first used, in a discrete-time system. The model was solved using LINGO 8.0 computer software. Next, a methodology was developed applying Hybrid Steady State Genetic Algorithm integrated to Linear Programming for solving the same model. After conducting tests comparing the MILP model with the new methodology and analyzing the results obtained, it was concluded it is concluded that the GA steady state hybrid-MILP methodology can contribute significantly to the problem of product transfer and storage in oil refineries. The results showed that the methodology is appropriate, taking into account the reduction of computational time, without much loss in quality solution that the new method resulted in a higher quality of problems solution and computational time.

a- Advantages

- Concepts are easy to understand
- Chances of getting optimal solution are more
- Less time required for some special applications
- Always an answer, answer gets better with time
- If large numbers of solutions of the problem exist but we have to find the best one, in this situation we can use genetic algorithm.

b- Disadvantages

- Genetic algorithms are very slow

- Genetic algorithm doesn't find exact solution but it always finds best solution
- Genetic algorithm is black art because parameter of genetic algorithm like elitism percentage, crossover, mutation rate etc is often just trial and error.
- More calculation
- Lower accuracy
- Longer time in calculating

c- Limitations

- Genetic algorithms don't provide guarantee of optimality of solution.
- Mutation rate should be low i.e. 0.5% - 1% assumed as best.
- The method of selection should be appropriate.
- Writing of fitness function must be accurate.
- Crossover rate should be 80% - 95%

Applications of genetic algorithm

- Genetic algorithm is used in robotics.
- Genetic Algorithm is applied in state assignment problem.
- Genetic Algorithms are most commonly used in optimization problems.
- Genetic algorithms have been used to plan the path which a robot arm takes by moving from one point to another.
- Genetic algorithms have been used to solve the TSP, which is a well-known combinatorial problem using novel crossover and packing strategies.
- Genetic algorithms are obviously very good approaches for multimodal optimization in which we have to find multiple optimum solutions.

Application of linear and integer programming problem

a- Applications of LPP

- Transportation problem is a type of linear programming problem that may be solved using a simplified version of the simplex technique.
- Linear programming techniques are also used in telecommunications. If there are many telephone calls being transmitted across a multi phone line network, linear programming is helpful in finding where it is necessary to build extra capacity.

b- Applications of IPP

The integer programming methods: pure integer programming, mixed integer programming, and zero-one programming techniques are used for the solution of following problems:

- Fixed charge problems
- Caterer problems
- Capital budgeting problems

Travelling salesman problems.

REFERENCE

A.O. Barboza, F.N. Junior, S.L.V. Bortolotti, R A de Souza, (2015), "Mixed integer linear programming and genetic algorithm applied to storage and transportation problems in an oil industry", System and Management, Vol. 10, pp 561-574.
C. Lewis, (2008), "Linear Programming: Theory and Applications" https://www.whitman.edu/lewis.
D.A. Coley, (1999), "An Introduction to Genetic Algorithm for scientists and Engineers", World Scientific, USA.
D.E. Goldberg, (1998), "Genetic Algorithm in Search, Optimization and Machine Learning", New York: Addison –Wesley.
E. Koenigsberg, (1961), "Some Industrial Applications of Linear Programming", journal of Operational Research Society, Vol. 12(2), pp 105-114.
F.L. Hitchcock, (1941), "The distribution of a product from several sources to numerous localities", MIT Journal of Mathematics and Physics 20:224–230.
F. Rothlauf, (2006), "Representation for Genetic and Evolutionary Algorithms", Springer, Netherlands, Second Edition.
G.B. Dantzig, (1963), "Linear programming and extensions", Princeton University Press and the Rand Corporation.
H. Holland, (1992), "Genetic Algorithms", Scientific American Journal.
Linear Programming, from Wikipedia, the free encyclopedia, 25/06/2017 https://en.m.wikipedia.org/wiki/linearprogramming.
L. Khachiyan, (1979), "A Polynomial-Time Algorithm for Solving Linear Programs", http://pubsonline.informs.org/doi/pdf/10.1287/moor.5.1.iv.
N. Karmarker, (1984), "A New Polynomial Time Algorithm for Linear Programming", Combinatorica, Vol 4, nr. 4, p. 373–395.
R.E. Gomory, (2010), "All Integer Programming Algorithms", 1st issued as IBM Research Center, Research Report RC-189, pp. 193-206.
S. Datta, C. Garai, C. Das, (2012), "Efficient Genetic Algorithm on linear programming problem for fittest Chromosome", journal of Global Research in computer Science, Vol 3, pp 1 – 7.
T.L. Feng, (2008), "A Genetic Algorithm for Linear Programming with Fuzzy Constraints", Journal of Information Science and Engineering, Vol.24, p. 801-817.