ISSN:0975-9646

Gaurang Panchal et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (2) , 2015, 1824-1827

Solving NP hard Problems using Genetic Algorithm

Gaurang Panchal & Devyani Panchal

U & P U Patel Department of Computer Engineering
Chandubhai S. Patel Institute of Technology
Changa, India

Abstract— The use of genetic algorithms was originally motivated by the astonishing success of these concepts in their biological counterparts. Despite this deferent approach, we can merely be seen as optimization methods, which are used in a wide range of applications. "Genetic algorithms (GA) are good at taking large, potentially huge search spaces and navigating them, looking for optimal combinations of things, solutions you would find difficult to accomplish." A genetic algorithm (GA) is an iterative search, optimization and adaptive machine learning technique premised on the principles of Natural selection. They are capable to finding solution to NP hard Problems. Genetic Algorithm based learning has promisingly showed results to a vast variety of function and problems. Travelling Salesman Problem, Tabu Search, and Transportation Problem is such classical problems for computation. This paper represents how to find optimal solution using various method of genetic algorithm. Advantages and disadvantages of this algorithm are reported and discussed.

Keywords-Crossover, Genetic Algorithm, Mutation, Random Population

I. INTRODUCTION

Genetic Algorithms are search, optimization and machine learning techniques based on the mechanics of Natural Selection and Natural Genetics. Genetic Algorithms (GA) are adaptive procedures of optimization and search that find solutions to problems by an evolutionary process inspired in the mechanisms of natural selection and genetic science. A genetic algorithm is a search method that functions analogously to an evolutionary process in a biological system. They are often used to find solutions to optimization problems [1,2]. GAs are Randomized search and optimization technique guided by the principle of natural genetic systems. Currently, these algorithms are being highly considered in those problems with complex solution spaces for those which we do not have good algorithms to solve them[3,4,5]. Genetic algorithms are algorithms that combine search algorithms with the genetics of nature. Past data and results are used to determine future results, in a 'survival of the fittest' kind of way. In a generation, the elements (the data represented as a string) that work the best move on to a new generation, with some mutations added in, just in case some important piece of information is lost through these changes. In a GA, a solution of our problem is called individual [6,7,8]. In essence, it consists of maintaining a population of a given number of individuals, each one of them characterized by a genetic code (genotype) that identifies it Univocally; thus an evolution of such population is simulated during the course of time, based on the apparition of new individuals resulting from crossovers, mutations and direct reproductions of the parents [9, 10]. An evaluation or objective function plays the role of the environment to distinguish in each generation that relatively good solutions reproduce, and that relatively bad solutions die, to be replaced by offspring of the good. Basically, we can say that a GA is based on the following components, for any type of application: a "genetic" representation of solutions to the problem; a way to create an initial population of solutions; an evaluation function to measure the fitness of any solution, and plays the role of the environment, in which the better solutions may have greater probability of survival; "genetic" operators that effect the composition of children during reproduction; value for the parameters that the algorithm uses to guide its evolution: population size, number of generations, crossing and mutation probabilities, etc[1].GA,NN and FL ,each of the technologies, in their own right and merit, has provided efficient solution to a wide range of problem [1-5]. Objective of the hybridization has been to overcome the weakness in one technology during its application, with the strengths of the others by appropriately integrate them. It investigating better methods of problem solving Hybrid systems has a tremendous potential to solve problem. Inappropriate use of technology can backfire. It has ability to locate the neighborhood of the optimal solution quicker than other conventional search strategies.

II. APPLICATION OF GENETIC ALGORITHM

Genetic algorithms (GA) are good at taking large, potentially huge search spaces and navigating them, looking for optimal combinations of things, solutions you would find difficult to accomplish. A genetic algorithm (GA) is an iterative search, optimization and adaptive machine learning technique premised on the principles of Natural selection. They are capable to finding solution to NP hard.

A. Different type of Application of Genetic Algorithm

As we are aware about some problems which take more time so solve. Such kind of problems can be solved using Genetic Algorithm e.g., Travelling Salesman Problem, Job shops Scheduling, Transportation.

The traveling salesman problem is of particular note because it is the classic example of non-deterministic polynomial (NP)

Complete problems that, so far, can only be solved in exponential time. Any problems can be classed as either solvable or unsolvable (such as the Halting Problem). The solvable problems they can be further subclasses as computationally complex or not. The Traveling Salesman Problem is the classic computationally complex problem. Imagine that you are a sales agent and you need to visit

potential clients in a certain number of cities using the shortest possible route. This problem can be solved using a computer. If there are n cities then the maximum number of possible itineraries between all the cities is (n – 1)! [5, 7, 9].An algorithm can be created which simply examines all the possible routes and comes up with the shortest one. However the main catch is that the amount of time required for the algorithm grows at an enormous rate as the number of cities increases. If there are 25 cities then the algorithm must examine 24! itineraries.24! is approximately 6.2•1023.Using a computer that can examine one million itineraries per second it would still take about 6.2•1023 / 106 = 6.2•1017 seconds to solve the problem. This is over 1.96•1010 years!

The Transportation Problem involves shipping a single commodity from suppliers to consumers to satisfy demand via the minimum cost. Assume that the supply equals the demand. There are m suppliers and n consumers. The cost of shipping one unit from a single supplier to each consumer is known. The problem is to find the best allocation of the commodity at the suppliers so that the demand can be satisfied and the lowest costs are incurred [11, 12, 13]. A matrix representation can be encoded into each chromosome. The suppliers are listed vertically and the consumers are listed horizontally. Element xij holds the number of commodity shipped from supplier i to consumer j.

During both the crossover and mutation operators, it is important to ensure that the amount of commodity being shipped remains constant since the amount of supply and Demand must remain equal.

Mutation involves randomly selecting a smaller sub-matrix consisting of a random number of rows and columns (greater than one) and then redistributing the values. The values are redistributed in such a way so that the sum of all values still remains constant (i.e. the same as before the mutation operator).

Biomimicry or biomimetic is the development of technologies inspired by designs in nature. Since GAs is inspired by the mechanisms of biological evolution, it makes sense that they could be used in the process of invention as well. GAs rely primarily on something called implicit parallelism (like to like), using mutation and selection in secondary roles toward a design solution.

GA programmers are working on applications that not only analyze the natural designs themselves for a return on how they work, but can also combine natural designs to create something entirely new that can have exciting applications. So the Genetic Algorithm is used widely for solving problems.

III. OVERVIEW OF GENETIC ALGORITHM

GAs was introduced as a computational analogy of adaptive systems. They are modeled loosely on the principles of the evolution via natural selection, employing a population of individuals that undergo selection in the presence of variation-inducing operators such as mutation and recombination (crossover). A fitness function is used to evaluate individuals, and reproductive success varies with fitness.

The Algorithms

a) Randomly generate an initial population M(0)
b) Compute and save the fitness u(m) for each individual m in the current population M(t)
c) Define selection probabilities p(m) for each individual m in M(t) so that p(m) is proportional to u(m)
d) Generate M(t+1) by probabilistically selecting individuals from M(t) to produce offspring via genetic operators
e) Repeat step 2 until satisfying solution is obtained.

The paradigm of GAs described above is usually the one applied to solving most of the problems presented to GAs. Though it might not find the best solution More often than not, it would come up with a partially optimal solution.

IV. GENETIC ALGORITHM

A. Parameter of Genetic Algorithm

Genetic Algorithm (GA) depends on some parameters like population size, maximum generation number, probability of crossover, a goal condition and probability of mutation. In our present study, we have taken the values of those parameters as follows: pop size = 20, crossover rate = 0.8, mutation rate = 0.10, maximum generation =100.

B. Encoding

Binary encoding is the most common, mainly because first works about GA used this type of encoding. In binary encoding every chromosome is a string of bits, 0 or 1.

Table 1. Binary Encoding

Chromosome A 10110010110010101110010 1

Chromosome B 11111110000011000001111 1

Binary encoding gives many possible chromosomes even with a small number of alleles. On the other hand, this encoding is often not natural for many problems and sometimes corrections must be made after crossover and/or mutation.

Permutation encoding can be used in ordering problems, such as travelling salesman problem or task ordering problem.

In Permutation encoding, every chromosome is a string of numbers, which represents number in a sequence.

Table 2 Permutation Encoding

Chromosome A : 1 5 3 2 6 4 7 9 8

1. for the entire parent chromosome do
   1a. toss the coin for crossover
      1a.1 if random number below probability of crossover

Chromosome B : 8 5 6 7 2 3 1 4 9

Example of chromosomes with permutation encoding Permutation encoding is only useful for ordering problems. Even for this problems for some types of crossover and mutation corrections must be made to leave the chromosome consistent (i.e. have real sequence in it).

In Value encoding, every chromosome is a string of some values. Values can be anything connected to problem, form numbers, real numbers or chars to some complicated objects.

Table 3 Value Encoding

Chromosome A: 1.2324 5.3243 0.4556 2.3293 2.4545

Chromosome B: ABDJEIFJDHDIERJFDLDFLFEGT

Chromosome C: (back), (back), (right), (forward), (left)

Value encoding is very good for some special problems. On the other hand, for this encoding is often necessary to develop some new crossover and mutation specific for the problem.

C. Initialization

We have taken binary encoding to generate initial random p population. Initial random population is depending upon the number of inputs. The initialization of any component of a chromosome can be done by random initialization, as the boundary of each component is not specified in the problem.

D. Crossover

Objective of this method is to apply the main operator of reproduction (i.e. Crossover) to the chromosomes of intermediate population. The name of the method signifies the work it has to do in program life cycle. Its basic work is to produce crossover between each paid of parent chromosomes. Normally we have higher crossover rate in application.

This is analogous to the crossover process in biological reproduction. The most general method is single point cross over. This method takes parent chromosomes from intermediate population, selected by roulette wheel or any other selection technique, and then at a particular random point, these chromosomes are crossed.

So, the region after the cross point in both chromosomes are interchanged. This crossover process is Probabilistic. The maximum probability of selection will decide, whether there should be crossover or not.

1a.1 (a) generate random number for crossover point
1a.1 (b) for bits after cross point
1a.1 (b). (i) Interchange bits between parent chromosomes

2. End

One Point Crossover

A crossover operator that randomly selects a crossover point within a chromosome then interchanges the two parent chromosomes at this point to produce two new offspring. Consider the following 2 parents which have been selected for crossover. The "|" symbol indicates the randomly chosen crossover-point.

Parent1:11001|010
Parent2:00100|111

after interchanging the parent chromosomes at the crossover point, the following offspring are produced:

Offspring1:11001|111
Offspring2: 00100|010

Two Point Crossover

A crossover operator that randomly selects two crossover points within a chromosome then interchanges the two parent chromosomes between these points to produce two new offspring.

Consider the following 2 parents which have been selected for crossover. The "|" symbols indicate the randomly chosen crossover-points.

Parent1:110|010|10
Parent2:001|001|11

after interchanging the parent chromosomes between the crossover points, the following offspring are produced:

Offspring1:110|001|10
Offspring2: 001|010|11

Uniform

A crossover operator that decides (with some probability – known as the mixing ratio) which parent will contribute each of the gene values in the offspring chromosomes. This allows the parent chromosomes to be mixed at the gene level rather than the segment level (as with one and two point crossover). For some problems, this additional flexibility outweighs the disadvantage of destroying building blocks.

Consider the following 2 parents which have been selected for crossover:

Parent1:11001010
Parent2:00100111

If the mixing ratio is 0.5, approximately half of the genes in the offspring will come from parent 1 and the other half will come from parent 2. Below is a possible set of offspring after

Uniform crossover:

Objective of this method is to randomly change one of the bits the parent chromosome bit string after crossover. The name of the method signifies the work it has to do in program life cycle. Its basic work is to randomly change information encoded in the chromosome bit string structure. This is same as biological mutation, in which sometimes error is generated during the reproduction

process there are mainly two different mutation techniques, which can be implemented in this method as its basic algorithm.

The Algorithm can be of the form:

1. for the entire parent chromosome do
    1a. generate a random number for probability of mutation
    1a.1 if number within range
    1a.1 (a) generate random mutation point
    1a.1 (b) mutate the bit

2. End

The most general method is single bit mutation. In this method, the coin is tossed to get the probability of mutation. If it lies below the specified range, the parent chromosome's bit string is mutated at random point. Generally, the probability of mutation is kept very low. This is a very good operator, as it exploits Global Optimum.

The great weakness of the conventional algorithm, in common with other heuristic approaches, there is no way to determine how far we are from the optimal solution. A genetic algorithm can be used to find a solution is much less time. Although it probably will not find the best solution, it can find a near perfect solution in less than a minute. We can conclude that the Genetic Algorithm is best search method as compared to the other conventional methods.

ACKNOWLEDGMENT

We avail this opportunity to acknowledge the academic interaction, exchange of views and participation with all those individual who have contributed toward this paper.

REFERENCES

[1] G. Panchal, A. Ganatra, Classification and Optimization to Evaluate the Fitness of an Algorithm. Lap Academic Publisher, Germany, 2012.

[2] A. Ganatra, G. Panchal,Y. Kosta, C. Gajjar, “Initial classification through back propagation in a neural network following optimization through GA to evaluate the fitness of an algorithm,” International Journal of Computer Science and Information Technology, vol. 3, no. 1, pp. 98–116, 2011.

[3] G. Panchal, A. Ganatra, Optimization of Neural Network Parameter using Genetic Algorithm. Lap Academic Publisher, Germany, 2012.

[4] G. Panchal and A. Ganatra, “Optimization of Neural Network Parameter using Genetic Algorithm,” 2008.

[5] D. P. Y. K. G. Panchal, A. Ganatra, “Performance analysis of classification techniques using different parameters,” Springer Verlag-Germany (Springer-LNCS), vol. 6411, 2010.

[6] G. Panchal, A. Ganatra, Y. Kosta, D. Panchal, “Searching most efficient neural network architecture using Akaikes information criterion (AIC),” International Journal of Computer Applications, vol. 1, no. 5, pp. 41–44, 2010.

[7] G. Panchal, “Forecasting Employee Retention Probability using Back Propagation Neural Network Algorithm,” IEEE 2010 Second International Conference on Machine Learning and Computing (ICMLC), Bangalore, India, pp. 248–251, 2010.

[8] G Panchal, A Ganatra, Y Kosta, D Panchal, “Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers,” International Journal of Computer Theory and Engineering, vol. 3, no. 2, pp. 332–337, 2011.

[9] G. Panchal, A. Ganatra, P. Shah, D. Panchal, “Determination of overlearning and over-fitting problem in back propagation neural network,” International Journal on Soft Computing, vol. 2, no. 2, pp. 40–51, 2011.

[10] G. Panchal, Y. Kosta, A. Ganatra,D. Panchal, “Electrical Load Forecasting Using Genetic Algorithm Based Back Propagation Network,” in 1st International Conference on Data Management, IMT Ghaziabad. MacMillan Publication, 2009.

[11] G. Panchal, A. Ganatra, P. Shah, Y. Kosta, “Unleashing Power of Artificial Intelligence for Network Intrusion Detection Problem,” International Journal of Engineering Science and Technology., vol. 2, no. 10, 2010.

[12] G. Panchal, A. Ganatra, D. Panchal, Y. Kosta, “Employee Retention Probability using Neural Network Based Back Propagation Algorithm,” IEEE Xplore, vol. 248, 2010.

[13] G. Panchal, D. Panchal, “Solving NP hard problem using Genetic Algorithm,” in National Women Conference, CITC, Changa.