An Overview of Genetic Algorithm and Modeling

Pushpendra Kumar Yadav[1], Dr.N.L.Prajapati[2]

[1] Research Scholar, Dept of Electronics and Communication, Bhagwant University, (Rajasthan) India
[2] Proffesor, Dept of Electronics and Communication, Indra Gandhi Engineering College, Sagar (M.P.) India

Abstract- Genetic Algorithm specially invented with for development of natural selection and genetic ideas. These are the basic algorithm used to generate solution for optimization problems with some natural evolution techniques which are inheritance, mutation, selection and crossover. The biologist explains that the life at the level of complexity that we observe could be evolved in the relatively short time. The father of the original Genetic Algorithm was John Holland who invented it in the year 1970's and Charles Darwin represent a method of a random search for a defined search space to solve a problem. Generally a population of string is used to encode candidate's solutions to an optimization problem represent for better solutions in binary 0s and 1s strings and this evolution usually starts from a population of randomly generated individuals in generations.

Index Terms- Basic steps, Criticism, Variants, Linkage Learning, Problem Domains, Evolutionary Algorithms.

I. INTRODUCTION

Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. These are generated for the evolution of natural selection and genetic ideas. Genetic Algorithms are better as compared to conventional AI in that it is more robust.AI system better work if the inputs changed slightly, or in the presence of reasonable noise. For searching a large state space, Multi modal state space or n dimensional surface.

II. BASIC STEPS OF GENETIC ALGORITHM

Genetic algorithm is placed in the knowledge based information system or evolutionary computing. Mainly two methods are there for Genetic algorithms:
1. A genetic representation of the solution domain,
2. A fitness function to evaluate the solution domain.

A genetic algorithm is a branch of evolutionary algorithm that is widely used.

To understand Evolution of Genetic Algorithms Justify different parameters are related to Genetic Algorithms.

This Table gives a list of different expressions, which are common in genetics with their equivalent in the framework of Genetic Algorithm's:

1	Natural Evolution	Genetic Algorithm
2	Genotype	Coded String
3	Phenotype	Uncoded Point
4	Chromosome	String
5	Gene	String Position
6	Allele	Value at a Certain Position
7	Fitness	Objective Function Value

Table 1: GA Expressions

The Figure below gives the Hierarchy of Knowledge based Information Systems:

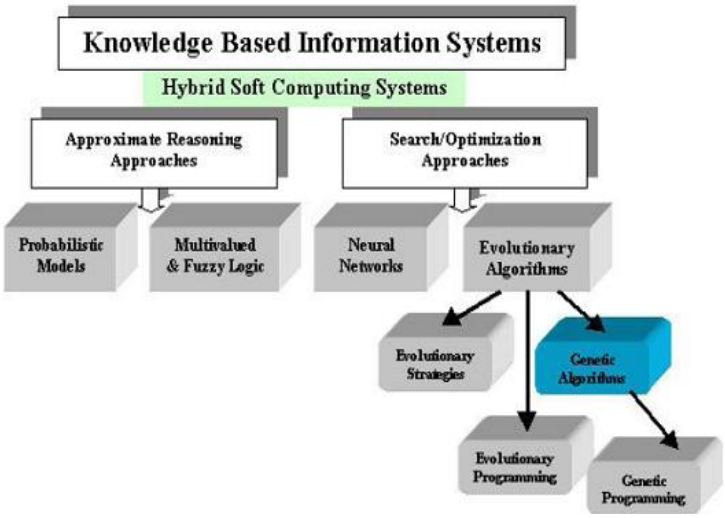

Figure 1: Steps of Genetic Algorithms

2.1 Simple generational genetic algorithm procedure:

a. Choose the initial population of individuals
b. Evaluate the fitness of each individual in that population
c. Repeat on this generation until termination (time limit, sufficient fitness achieved, etc.):
d. Select the best-fit individuals for reproduction
e. Breed new individuals through crossover and mutation operations to give birth to offspring
f. Evaluate the individual fitness of new individuals
g. Replace least-fit population with new individuals

2.2 Basic structure of a genetic algorithm is given hereunder:

Algorithm
t: = 0;
Compute initial population B0;
WHILE stopping condition not fulfilled DO

BEGIN
Select individuals for reproduction;
Create offspring's by crossing individuals;
Eventually mutate some individuals;
Compute new generation
END

As obvious from the above algorithm, the transition from one generation to the next consists of four basic components:

a. Selection: Mechanism for selecting individuals (strings) for reproduction according to their fitness (objective function value).
b. Crossover: Method of merging the genetic information of two individuals; if the coding is chosen properly, two good parents produces good children.
c. Mutation: In real evolution, the genetic material can by changed randomly by improper reproduction or other deformations of genes, e.g. by gamma radiation. In genetic algorithms, mutation can be realized as a random deformation of the strings with a certain probability. The positive effect is preservation of genetic diversity and, as an effect, that local maxima can be avoided.
d. Sampling: This is the Procedure which computes a new generation from the previous one and its off springs. Compared with traditional continuous optimization methods, such as Newton or gradient descent methods, we can state the following significant differences:

III. The Building Block Hypothesis

We can easily implement Genetic Algorithms but their behavior is difficult to understand. It is difficult to understand why these algorithms frequently succeed at generating solutions of high fitness when applied to practical problems. The building block hypothesis (BBH) consists of:

a. A description of a heuristic that performs adaptation by identifying and recombining "building blocks", i.e. low order, low defining-length schemata with above average fitness.
b. A hypothesis that a genetic algorithm performs adaptation by implicitly and efficiently implementing this heuristic.

IV. Observations

There are several general observations about the generation of solutions for genetic algorithms:

a. Selection is clearly an important genetic operator, but opinion is divided over the importance of crossover versus mutation. One says crossover is the most important, while mutation is only necessary to ensure that potential solutions are not lost. Others says that crossover in a largely uniform population only serves to propagate innovations originally found by mutation, and in a non-uniform population crossover is nearly always equivalent to a very large mutation.

b. With all current machine learning problems it is worth tuning the parameters such as mutation probability, crossover probability and population size to find reasonable settings for the problem class being worked on. A very small mutation rate may lead to genetic drift. A recombination rate that is too high may lead to premature convergence of the genetic algorithm. A mutation rate that is too high may lead to loss of good solutions unless there is elitist selection. Often, GAs can rapidly locate *good* solutions, even for large search spaces. The same is of course also true for evolution strategies and evolutionary programming.

V. Criticisms

There are several criticisms of the use of a genetic algorithm compared to alternative optimization algorithms:

a. Repeated fitness function evaluation for complex problems are often the most prohibitive and limiting segment of artificial evolutionary algorithms. Finding the optimal solution to complex high dimensional, multimodal problems often requires very expensive fitness function evaluations. In real world problems such as structural optimization problems, one single function evaluation may require several hours to several days of complete simulation. Typical optimization methods cannot deal with such types of problem. In this case, it may be necessary to forgo an exact evaluation and use an approximated fitness that is computationally efficient.

b. Operating on dynamic data sets is difficult, as genomes begin to converge early on towards solutions which may no longer be valid for later data. Several methods have been proposed to remedy this by increasing genetic diversity somehow and preventing early convergence, either by increasing the probability of mutation when the solution quality drops.

c. For specific optimization problems and problem instances, other optimization algorithms may find better solutions than genetic algorithms (given the same amount of computation time). Alternative and complementary algorithms include evolution strategies, evolutionary programming, simulated annealing, Gaussian adaptation, hill climbing, and swarm intelligence (e.g.: ant colony optimization, particle swarm optimization) and methods based on integer linear programming.

VI. Variants

a. The simplest algorithm represents each chromosome as a bit string. Typically, numeric parameters can be represented by integers, though it is possible to use floating point representations. The floating point representation is natural to evolution strategies and evolutionary programming. The notion of real-valued genetic algorithms has been offered but is really a misnomer because it does not really represent the building block theory that was proposed by John Henry

Holland in the 1970s. This theory is not without support though, based on theoretical and experimental results.

b. When bit-string representations of integers are used, Gray coding is often employed. In this way, small changes in the integer can be readily effected through mutations or crossovers. This has been found to help prevent premature convergence at so called *Hamming walls*, in which too many simultaneous mutations (or crossover events) must occur in order to change the chromosome to a better solution.

c. Genetic algorithms with adaptive parameters or adaptive genetic algorithms are another significant and promising variant of genetic algorithms. The probabilities of crossover and mutation greatly determine the degree of solution accuracy and the convergence speed that genetic algorithms can obtain. It can be quite effective to combine GA with other optimization methods. GA tends to be quite good at finding generally good global solutions, but quite inefficient at finding the last few mutations to find the absolute optimum. Other techniques (such as simple hill climbing) are quite efficient at finding absolute optimum in a limited region. Alternating GA and hill climbing can improve the efficiency of GA while overcoming the lack of robustness of hill climbing

VII. Linkage-learning

A number of variations have been developed to attempt to improve performance of GAs on problems with a high degree of fitness epistasis, i.e. where the fitness of a solution consists of interacting subsets of its variables. Such algorithms aim to learn (before exploiting) these beneficial interactions. As such, they are aligned with the Building Block Hypothesis in adaptively reducing disruptive recombination. Prominent examples of this approach include the mGA,\ GEMGA and LLGA.

VIII. Problem domains

Problems which appear to be particularly appropriate for solution by genetic algorithms include timetabling and scheduling problems, and many scheduling software packages are based on GAs. GAs has also been applied to engineering. Genetic algorithms are often applied as an approach to solve global optimization problems. Examples of problems solved by genetic algorithms include: mirrors designed to funnel sunlight to a solar collector, antennae designed to pick up radio signals in space, and walking methods for computer figures. Many of their solutions have been highly effective, unlike anything a human engineer would have produced, and inscrutable as to how they arrived at that solution.

IX. Evolutionary Algorithms

An evolutionary algorithm is a sub-field of evolutionary computing.

a. Strategies: Evolution strategies evolve individuals by means of mutation and intermediate or discrete recombination. ES algorithms are designed particularly to solve problems in the real-value domain.

b. Evolutionary Programming: Evolutionary programming (EP) involves populations of solutions with primarily mutation and selection and arbitrary representations.

c. Gene Expression Programming: Gene expression programming (GEP) uses populations of computer programs. These complex computer programs are encoded in simpler linear chromosomes of fixed length, which are afterwards expressed as expression trees.

d. Genetic Programming: Genetic programming (GP) is a technique in which computer programs, rather than function parameters, are optimized. Genetic programming often uses tree-based internal data structures to represent the computer programs for adaptation instead of the list structures typical of genetic algorithms.

e. Grouping Genetic Algorithm: Grouping genetic algorithm (GGA) is an evolution of the GA where the focus is shifted from individual items, like in classical GAs, to groups or subset of items. Basically clusteringor *partitioning* problems where a set of items must be split into disjoint group of items in an optimal way would better be achieved by making characteristics of the groups of items equivalent to genes.

f. Interactive Evolutionary Algorithms: Interactive evolutionary algorithms are evolutionary algorithms that use human evaluation. They are usually applied to domains where it is hard to design a computational fitness function.

X. Conclusion and Future Work

Genetic Algorithms appears better in finding areas of interest in a complex, real world scene. Genetic Algorithms are adaptive to their environments, as this type of method is a platform appearing in the changing environment. In Present these algorithms are more applicable. Several improvements must be made in order that GAs could be more generally applicable. Grey coding the field would greatly improve the mutation operation while combing segmentation with recognition so that the interested object could be evaluated at once and timing improvement could be done by utilizing the implicit parallelization of multiple independent generations evolving at the same time.

References

[1] Eiben, A. E. et al (1994). "Genetic algorithms with multi-parent recombination". PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature: 78–87. ISBN 3-540-58484-6.

[2] Ting, Chuan-Kang (2005). "On the Mean Convergence Time of Multi-parent Genetic Algorithms without Selection". Advances in Artificial Life: 403–412. ISBN 978-3-540-28848-0.

[3] Akbari, Ziarati (2010). "A multilevel evolutionary algorithm for optimizing numerical functions" IJIEC 2 (2011): 419–43

[4] [G. Harik. Learning linkage to efficiently solve problems of bounded difficulty using genetic algorithms. PhD thesis, Dept. Computer Science, University of Michigan, Ann Arbour, 1997

[5] Goldberg, David E. (1989). Genetic Algorithms in Search Optimization and Machine Learning. Addison Wesley. P.41 ISBN0-201-15767-5.

[6] Wolpert, D.H., Macready, W.G., 1995. No Free Lunch Theorems for Optimisation. Santa Fe Institute, SFI-TR-05-010, Santa Fe.

[7] Doctoral Dissertation, University of Michigan, Dissertation Abstracts International 28(12) Cavicchio: Adaptive search using simulated evolution, unpublished doctoral dissertation, University of Michigan, Ann Arbor

[8] DeReffye, Edelin, Francon, Jaeger,. Puech: Plant Models faithful to botanical structure and development, ACM Computer Graphics SIGGRAPH Proc., Vol 22(4), (1988)

[9] Gervautz, Traxler: Representation and Realistic Rendering of Natural Scenes with Cyclic CSG graphs, accepted for publication in Visual Computer, 1995

[10] Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning, 1989, Addison-Wesley Publishing Co.

[11] Holland: Adaptation in Natural and Artificial Systems, Ann Arbor, Mi: University of Michigan Press, 1975

[12] De Jong: An Analysis of the behavior of a class of genetic adaptive systems, Doctoral Dissertation, University of Michigan, Dissertation Abstracts International 36(10)

AUTHORS

First Author – Pushpendra Kumar Yadav, Research Scholar, Dept of Electronics and Communication, Bhagwant University,(Rajasthan)India, Email- pushpendraecmit@rediffmail.com

Second Author – Dr.N.L.Prajapati, Proffesor, Dept of Electronics and Communication, Indra Gandhi Engineering College, Sagar (M.P.) India, Email—praja05@rediffmail.com