Genetic Algorithm – Survey Paper

Anita Thengade
Assistant Professor, Department of Computer Engineering, MAEER'S MITCOE Pune, India

Rucha Dondal
ME in Computer Engineering(SEM-I) MAEER'S MITCOE Pune, India

Abstract
This paper provides an introduction of Genetic Algorithm, its basic functionality. The basic functionality of Genetic Algorithm include various steps such as selection, crossover, mutation. This paper also focuses on the comparison of Genetic Algorithm with other problem solving technique. The details of labs that basically concentrate on the research and development of Genetic Algorithm is also included. The details of labs include the various projects that are carried out on Genetic Algorithm.

Keywords
Selection, Crossover, Mutation, problem-solving technique.

1. INTRODUCTION
Genetic Algorithm is a programming technique who forms its basis from the biological evolution. [6] Genetic Algorithm is basically used as a problem solving strategy in order to provide with a optimal solution. They are the best way to solve the problem for which little is known. They will work well in any search space because they form a very general algorithm. The only thing to be known is what the particular situation is where the solution performs very well, and a genetic algorithm will generate a high quality solution. Genetic algorithms use the principles of selection and evolution to produce several solutions to a given problem.

- **Individual** - Any possible solution
- **Population** - Group of all *individuals*
- **Search Space** - All possible solutions to the problem
- **Chromosome** - Blueprint for an *individual*
- **Trait** - Possible aspect of an *individual*
- **Allele** - Possible settings for a *trait*
- **Locus** - The position of a *gene* on the *chromosome*
- **Genome** - Collection of all *chromosomes* for an *individual*

The input to the GA is a set of potential solutions to that problem, encoded in some fashion, and a metric called a *fitness function* that allows each candidate to be quantitatively evaluated. These candidates may be solutions already known to work, with the aim of the GA being to improve them, but more often they are generated at random.

The GA then evaluates each candidate according to the fitness function activity; the candidate with good fitness has high chances to get selected than the one with average fitness. Various functions is basically used to test the fitness of any particular individual. These individuals with high fitness can be termed as promising candidates.

These promising candidates are kept and allowed to reproduce. From them multiple copies are made, but the copies are not perfect; random changes are introduced during the copying process. These digital offspring then go on to the next generation, forming a new pool of candidate solutions, and are subjected to a second round of fitness evaluation. Those candidate solutions which were worsened, or made no better, by the changes to their code are again deleted; but again, purely by chance, the random variations introduced into the population may have improved some individuals, making them into better, more complete or more efficient solutions to the problem at hand. Again these winning individuals are selected and copied over into the next generation with random changes, and the process repeats. The expectation is that the average fitness of the population will increase each round, and so by repeating this process for hundreds or thousands of rounds, very good solutions to the problem can be discovered.

[1]Genetic algorithms have proven to be an enormously powerful and successful problem-solving strategy. Genetic algorithms have been used in a wide variety of fields to find solutions to problems that are more difficult than those faced by human designers. Thus, the solutions they come up with are often more efficient, more elegant, or more complex than anything comparable a human engineer would produce.

2. BASIC FUNCTIONALITY OF GENETIC ALGORITHM

2.1 Selection
[2] There are many different techniques which a genetic algorithm can use to select the individuals to be copied over into the next generation.

Elitist selection: The fittest members of each generation are guaranteed to be selected.

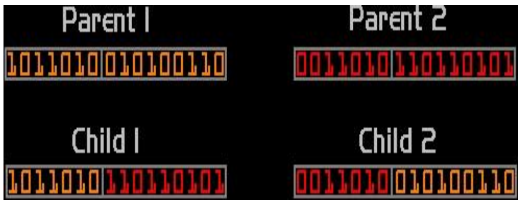
Fitness-proportionate selection: More fit individuals are more likely, but not certain, to be selected.

Roulette-wheel selection: A form of fitness-proportionate selection in which the chance of an individual's being selected is proportional to the amount by which its fitness is greater or less than its competitors' fitness

Scaling selection: As the average fitness of the population increases, the strength of the selective pressure also increases and the fitness function becomes more discriminating. This method can be helpful in making the best selection later on when all individuals have relatively high fitness and only small differences in fitness distinguish one from another.

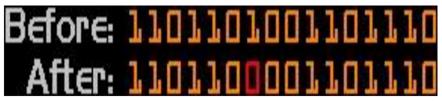
Tournament selection: Subgroups of individuals are chosen from the larger population, and members of each subgroup compete against each other. Only one individual from each subgroup is chosen to reproduce.

2.2 Crossover
Once the individuals have been selected the next thing is to produce the offspring. [2] The most common solution for this is something called crossover, and while there are many different kinds of crossover, the most common type is single point crossover. In single point crossover, chooses a locus at which you swap the remaining alleles from one parent to the other. This is complex and is best understood visually.



The children take one section of the chromosome from each parent. The point at which the chromosome is broken depends on the randomly selected crossover point. This particular method is called single point crossover because only one crossover point exists. Sometimes only child 1 or child 2 is created, but oftentimes both offspring are created and put into the new population. Crossover does not always occur, however. Sometimes, based on a set probability, no crossover occurs and the parents are copied directly to the new population. The probability of crossover occurring is usually 60% to 70%.

2.3 Mutation
After selection and crossover, we get new population full of individuals. Some are directly copied, and others are produced by crossover. In order to ensure that the individuals are not all exactly the same, you allow for a small chance of mutation. You loop through all the alleles of all the individuals, and if that allele is selected for mutation, you can either change it by a small amount or replace it with a new value.[3] The probability of mutation is usually between 1 and 2 tenths of a percent. A visual for mutation is shown below.



Mutation is fairly simple. We just change the selected alleles based on what you feel is necessary and move on. Mutation is, however, vital to ensuring genetic diversity within the population.

3. COMPARISON WITH OTHER PROBLEM SOLVING TECHNIQUES
This section deals with the three aspects of GAs. Firstly we specified the different issues and challenges that are faced by the GA.[4] Then we have elaborated the difference between the GA and the Heuristic Search and finally we have provided the detailed comparison of GA with the various search techniques such as the Hill Climbing and Simulating Annealing.

3.1 Issues Related to the Genetic Algorithm
1. Certain optimization problems (they are called variant problems) cannot be solved by means of genetic algorithms. This occurs due to poorly known fitness functions which generate bad chromosome blocks in spite of the fact that only good chromosome blocks cross-over..
2. There is no absolute assurance that a genetic algorithm will find a global optimum. It happens very often when the populations have a lot of subjects.
3. In case of Clinical decision support system GA faces lack of transparency that is used for the decision support systems making it undesirable for physicians. The main challenge in using genetic algorithms is in defining the fitness criteria. In order to use a genetic algorithm, there must be many components such as multiple drugs, symptoms, treatment therapy and so on must be available in order to solve a problem.
4. In case of research study which investigated the possibility of automating parameter selection for an EEG-based P300-driven Brain-Computer Interface (BCI) Genetic Algorithm required lengthy execution times which render it infeasible for online utilization. The GA method was subsequently replaced by the less execution time-intensive N-fold cross-validation (NFCV) for the meta-optimization of feature extraction and pre-processing parameters using Fisher's Linear Discriminant Analysis (FLDA).
5. GA also faces Scalability Issues of Exchanging Building Blocks.
6. Like other artificial intelligence techniques, the genetic algorithm cannot assure constant optimization response times. Even more, the difference between the shortest and the longest optimization response time is much larger than with conventional gradient methods. This unfortunate genetic algorithm property limits the genetic algorithms' use in real time applications.
7. Genetic algorithm applications in controls which are performed in real time are limited because of random solutions and convergence, in other words this means that the entire population is improving, but this could not be said for an individual within this population. Therefore, it is unreasonable to use genetic algorithms for on-line controls in real systems without testing them first on a simulation model.

3.2 Comparison between Genetic Algorithm and Heuristic Search.

Genetic Algorithm	Heuristic Search
1)Genetic Algorithm are typically applied to problems that require only the location of a vertex that satisfies the search	1) Heuristic search often require the construction of a path through the graph.
2) A problem addressed by Genetic Algorithm does not specify a starting space or/and goal state.	2) A problem addressed by Heuristic Search often specify a starting space or/and goal state.
3) Genetic Algorithm is often applied to problems that do not specify how an acceptable solution can be recognized.	3) Heuristic Search is often applied to problems that do specify how an acceptable solution can be recognized.
4) The graph searched via heuristic algorithm does not determine predefined connectivity.	4) The graphs searched via heuristic algorithms often have some predefined connectivity determined.

	E.g.: the mechanical properties of a puzzle.
5) Genetic Algorithm navigates on several graphs.	5) State space search algorithm tends to navigate on a single graph.

Table 1- Comparison of Genetic Algorithm with Heuristic Search

3.3 Comparison of Genetic Algorithm with Hill Climbing and Simulating Annealing

Parameters	Genetic Algorithm	Hill Climbing	Simulated Annealing
Functionality	Genetic Algorithm belong to a family of computational models inspired by evolution. These algorithms encode a potential solution to a specific problem on a simple chromosome-like data structure and apply recombination operators to these structures as to preserve critical information.	Hill climbing is a mathematical optimization technique which belongs to the family of local search. It is an iterative algorithm that starts with an arbitrary solution to a problem, then attempts to find a better solution by incrementally changing a single element of the solution.	Simulated annealing (SA) is a generic probabilistic metaheuristic for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space.
Performance	Genetic Algorithm are faster than the hill climbing as it implements implicit parallelism wherein each new string is completely independent of the previous one, so each new string is given independently to the new schema region.	Moves in space of string by single bit mutation from an original string, so each new sample has all but one of the same bits as the previous sample.	Simulating Annealing performance is slower as compared to GA.
Representation	Bit-String representation is critical.	MoveSet design is critical.	MoveSet design is critical.
Muilti-dimentional solving	[5] Supports the muilti-dimentional approach	Does not support the muilti-dimentional approach	[6]Does not support the multi dimensional approach.
Technique preferred	Global Optimum	Local Optimum	Global Optimum
Distribution of population is diverse and large	GA makes a large, useful and structured jump even after partial convergence.	Hill Climbing is not good as compared to GA when the distribution of population is diverse and large.	Simulating Annealing is not so good as compared to GA.
Least diverse population	GA does not gain benefit in least diverse population.	Hill climbing search operate fruitfully produces good result.	Simulating Annealing does not gain benefit in least diverse population
Search via Sampling i.e. Blind search.	GAs achieves much of their breadth by ignoring information except that concerning payoff.	Rely heavily on such information and in problem where the necessary information is not available or difficult to obtain, these technique breaks down.	Simulating Annealing too rely heavily on such information.
Search Mechanism	GA performs a multi-directional search by maintaining a population of potential solution and encourages information exchange between these directions.	HC uses a iterative improvement technique, which is applied to a single point in the search space	Simulating Annealing is used when the search space is discrete.
Examples	Acoustics, Aerospace engineering, Astronomy and astrophysics, Chemistry, Electrical engineering, Financial markets, Game playing, Geophysics, Materials engineering.	The travelling salesman problem, the eight-queens problem, circuit design.	Simulated annealing is often used for engineering design applications such as determining the physical layout of components on a computer chip

Table 2 – The basic comparison of Genetic Algorithm with other problem solving method.

4. GENETIC ALGORITHM RESEARCH LABS

On basis of the survey there are various labs available that basically focuses on the research that are carried out on the Genetic Algorithm

4.1 Kanpur Genetic Algorithms Laboratory

(KanGAL) is a research laboratory at Indian Institute of Technology Kanpur dedicated to pursuing research and industrial collaboration in the areas of genetic algorithms, fuzzy logic controllers, bioinformatics and neural networks. Fundamental research and application to engineering problems are primary focus of KanGAL.

KanGAL was established in 1997. Since then the lab has produced a number of doctoral theses and masters theses related to genetic algorithms, fuzzy logic controllers, and neural networks. A number of undergraduate students have also been trained in these areas. Besides fundamental research, KanGAL is also active in collaborating with industries and is particularly interested in using soft computing techniques for industrial problem solving.

4.2 Genetic Algorithm Research Colorado State University

The GENITOR group is a research group within the **Colorado State AI Lab (CSAIL)** in the Computer Science Department at Colorado State University. The group focuses on the theory and applications of Genetic Algorithms, Evolutionary Computation and Search.

The group is headed by **Dr. Darrell Whitley** and is made up of several students at the PhD, Masters and Undergraduate levels. Dr. Whitley is also director of the **Colorado State AI Lab (CSAIL)** which is made up of 5 faculty and approximately 15 graduate researchers. The GENITOR group was named for the **Genitor** genetic algorithm developed by Whitley et. al. in 1989.

Some of the project areas in which GENITOR group members have worked or are currently working are:

- Boolean Satisfiability.
- Evaluating Test Functions for Evolutionary Algorithms
- Evolving Neural Networks
- Infinite Population Models
- Parameter Optimization
- Representation Issues for Genetic Algorithms
- Scheduling
- Seismic Applications
- Travelling Salesman Problems
- Walsh Analysis

4.3 GARAGe

The MSU **Genetic Algorithms Research and Applications Group** (the GARAGe) is a multi-disciplinary unit interested in the application of genetic algorithms, genetic programming and other forms of evolutionary computation to real-world problems, as well as fundamental research on GA and GP. They have recently done a number of interesting projects, both in terms of GA/GP fundamental research and in GA/GP applications. Projects involving Prof. Goodman include:

- Goodman's new NSF Cyber-enabled Discovery and Innovation grant (Sept., 2009) with Profs. Johannes Bauer, a telecomm policy expert, and Kurt DeMaagd, an expert in internet growth and modeling, will study "Improving Governance of Next-Generation ICT Infrastructure." This three-year grant will put three graduate research assistants (from Telecommunications, Information Studies and Media and from Electrical and Computer Engineering, working as a team developing agent-based models of enterprises or sectors, classes of customers, and non-profit and government agencies, geographically distributed, and using evolutionary computation both to parameterize them and to evolve governance structures that drive their behaviors in desired directions.
- Research on sustainable and scalable evolutionary methods, resulting in development of the **Hierarchical Fair Competition** principle.
- Automated design, including research on composite material design and multi-objective design of automotive components for crashworthiness, weight savings, and other characteristics.
- Automated design of mechatronic systems using bond graphs and genetic programming (NSF). Current funding (2007-2010) is from the Danish Research Council to Technical University of Denmark, where Assoc. Prof. Zhun Fan leads the project. New Ph.D. student Jean-Francois Dupuis worked on the problem in the GARAGe during 2007-2008, with Goodman.
- Multi-objective "compatible" control algorithms, "MOCC," in which evolutionary methods are used to evolve controllers for such systems as commercial greenhouses, minimizing energy usage while keeping environmental parameters within bounds around the optimum. This research is joint with visiting professor Lihong Xu, of Tongji University, Shanghai, and a series of his Ph.D. students (2006-2010).
- Evolutionary improvement of medical diagnosis through pattern classification methods applied to high-dimensionality sets of laboratory test data, with Visiting Prof. Ping Wu, East China Normal University, 2008-2009.
- Recognition of breast tumors in microwave imaging data, using evolutionary methods for solving the inverse problem from a novel microwave imaging system being developed by Visiting Prof. Meng Yao, East China Normal University, 2009-2010.
- Agent-based modeling, with parameter optimization via evolutionary methods, of an emergency urban evacuation scenario, with undergrad Professorial Assistant Matt Durak, for use at the Academy of Critical Incident Analysis, John Jay College, City University of New York.
- GA operator modeling, with Visiting Prof. Zhenhua Li, China University of Geosciences (Wuhan) (2006-2007)

- Mathematical modeling, by Bulent Buyukbozkirli, of genetic algorithm behavior, aimed at gaining insight useful for configuring genetic algorithms to solve particular problems
- Parallelization of GAs/GPs including use of hierarchical decomposition of problem domains and heterogeneous design spaces using concepts such as iiga, the injection island GA
- Nesting of irregular shapes using feature matching and GAs
- Multiple population topologies and interchange methodologies
- Mobile communications infrastructure optimization
- Scheduling applications, including job-shop scheduling
- Configuration applications, particularly physics applications of optimal molecule configurations for particular systems like C60 (buckyballs)
- Protein folding and protein/ligand docking

5. CONCLUSION

If the conception of a computer algorithms being based on the evolutionary of organism is surprising, the extensiveness with which this algorithms is applied in so many areas is no less than astonishing. These applications, be they commercial, educational and scientific, are increasingly dependent on this algorithms, the Genetic Algorithms. Its usefulness and gracefulness of solving problems has made it the more favorite choice among the traditional methods, namely gradient search, random search and others. GAs is very helpful when the developer does not have precise domain expertise, because GAs possesses the ability to explore and learn from their domain.

In this paper we placed more emphasis on describing the basic of the Genetic Algorithm, its functionality. The various issues that are faced by Genetic Algorithm are also included in this paper. We have also discussed the detailed comparison of GAs with Heuristic Search and also with other searching technique such as Simulating Annealing and Hill Climbing. It also includes the various Labs that mainly concentrate on the different research that are carried out on GAs.

In future work we basically concentrate on the various issues related to the GA, what all strategies are needed to be implemented in order to overcome these issues so as to make GA even more powerful and efficient.

6. REFERENCES

[1] "Adaptive Learning: Fly the Brainy Skies." Wired, vol.10, no.3 (March 2002).

[2] Andre, David and Astro Teller. "Evolving team Darwin United." In RoboCup-98: Robot Soccer World Cup II, Minoru Asada and Hiroaki Kitano (eds). Lecture Notes in Computer Science, vol.1604, p.346-352. Springer-Verlag, 1999.

[3] Coello, Carlos. "An updated survey of GA-based multiobjective optimization techniques." ACM Computing Surveys, vol.32, no.2, p.109-143 (June 2000).

[4] Fonseca, Carlos and Peter Fleming. "An overview of evolutionary algorithms in multiobjective optimization." Evolutionary Computation, vol.3, no.1, p.1-16 (1995).

[5] J. H. Holland. Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor, MI, 1975.

[6] Mitchell, Melanie. An Introduction to Genetic Algorithms. MIT Press, 1996.