A Genetic Algorithms Analysis for Optimization Solutions of Complex Problems

Er. Seema Rani,
Assistant Professor
Computer Science and Engineering,
CDLSIET, Sirsa, India.

Abstract: Regardless of how simple or complicated the problem, today's world demands optimal best practices for clever problem solving across all industries. The goal of research and development is to increase the intelligence and efficiency of hardware and software. Here is where artificial intelligence comes into play, helping to create effective and ideal searching algorithm solutions. We require a metric to distinguish between good and bad options to arrive at the answers. A statistical model or simulation could serve as an objective metric, or it could be subjective and rely on our preference for better answers. A fitness function identifies the optimal solution for a given problem, and the GA uses this information to direct the evolution of optimal solutions. A most popular heuristic search technique in AI is genetic algorithm. Goal of a genetic algorithm (GA) is to use heredity, mutation, selection, and other processes to best possible solution. It has been demonstrated that effective & neutral optimization method are genetic algorithms. Study demonstrates the applications of genetic algorithms in multiple fields and how they may be integrated with other techniques and approaches to create optimal solutions and boost the retrieval system's computation time.

***Index Terms* -** Genetic Algorithm, Optimal Solution, Fitness function, GA operators, Meta-heuristics.

I. INTRODUCTION

Genetic algorithms are search and optimization methods that were first put forth by John Holland in 1970 and are predicated on the concepts of natural evolution. Genetic algorithms simulate species evolution through natural selection, hence implementing optimization methodologies as well. GAs differs from conventional search engines in that they have a parallel population-based search with stochastic crossover, mutation, and selection of several individual solutions. Genetic algorithms have this particular set of components, whereas other techniques just have a few [1]. Genetic algorithms offer many benefits and drawbacks and are applied in various fields. Though the outcomes are typically rather near to the global optimum, GA does not guarantee. The solutions are not contained by local optima because they are thought to be probabilistic in nature. Typically, a genetic algorithm has two phases. To develop the next generation, one person must be selected initially, and then the selected individual must be modified utilizing crossover and mutation processes [2]. The selection mechanism determines which individuals are chosen for reproduction and how many children each chosen individual produces. The fundamental idea of selection strategy is that a person's goodness increases the likelihood that they will become a parent. This research paper aims to find the parametric values of the GA that yield the most optimum solution (maximum generation, optimum population, optimum crossover rate, and optimum mutation rate) so that we can better understand the effectiveness of this algorithm and the strategies we could use to enhance performance while searching for the optimal solution. The research paper is organized as, First section explain introduction. Second section provides related work. Third section describes genetic algorithm, Fourth section explain GA operators. Section V defines the fitness function. Next section tells about different barriers and last section concludes the paper.

II RELATED WORK

The Fast Genetic K-means Algorithm (FGKA) is an enhanced variant of GKA [3]. The suggested GA was superior to GKA in a number of ways. Experiments showed that while the K-means algorithm might give a optimum solution but the global optimum solution is always reached by the FGKA and GKA algorithms. By using selection, crossover, and mutation operators, FGKA first initialize the population then creates the subsequent population. The population continues to evolve until a termination state is satisfied. While illegitimate strings are allowed in initial phase of FGKA, the cluster variations (TWCVs) were defined as infinite ($+\infty$) in order to identify them as the least desired alternatives. Allowing unlawful strings reduced the overhead associated with them during the evolution process, which enhanced time performance.

In order to expedite the fitness evaluation process, a GA-based clustering technique was presented [4]. This technique chooses cluster from data set and saves the distances between every pair of data points. More efficient operators for selection, crossover, mutation were added, and variable numbers of cluster centers are encoded using binary data as opposed to string representation.
A unique method of [5] using a genetic algorithm in tandem with concurrent indexing of picture color and feature extraction has been put into practice. Its primary function is image-to-image matching, and retrieving still images is the intended use. The GA provides the assessment criteria, which have been effectively used to assess effectiveness of image retrieval process.

A new process like conventional GA is presented in [6], which makes several improvements to the conventional GA algorithm. A genetic selection technique was used, which reduced the likelihood of becoming stuck in a local optimum. The novel algorithm has a lower complexity and expands the searching space as compared to the conventional genetic algorithm. It is determined by examining the benchmark function optimization testing results that the novel approach outperforms the conventional genetic algorithm in terms of optimization precision.

Groundwater supply management has found AI and GAs to be helpful. They suggested using GAs to fit model parameters in order to maximize groundwater treatment pumping sites and schedules. The complicated reaction functions inside the GA were subsequently modeled by combining the GA with a neural network (NN) [7]. Additionally, oceanographic experiment design has used evolutionary approaches. For oceanographic experiment, a genetic algorithm is more accurate than a problem-specific technique and faster than simulated annealing. For the purpose of detecting a group of sensors in water, evolutionary programming strategy proved to be more reliable than traditional methods.

III GENETIC ALGORITHM

Genetic algorithms, or GA, are a population-based heuristic approach that mimics the process of natural evolution [8]. Three operators—crossover, mutation, and selection—are applied to each individual with different probabilities during each generation. Through genetic processes like crossover and mutation, a GA's iterative operations modify one population of chromosomes (solution candidates) to create a new population.

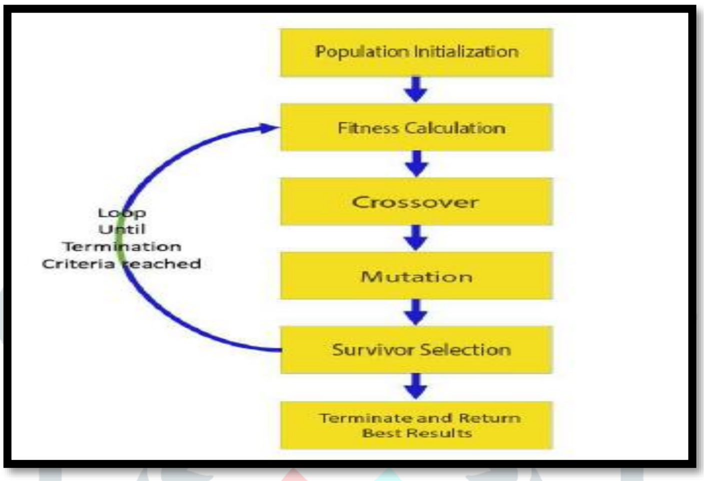

Figure 1: Genetic Algorithm Flow-Chart

A computer program called GAs mimics the evolution and heredity of living things. Since GAs are multi-point search techniques, they can also be used for multi-modal objective functions and still produce an optimal solution. Steps of genetic algorithm are:

1. Produce random population of n chromosomes [Start].
2. Assess each chromosome x's fitness, f(x), within the population [Fitness].
3. Once the new population is complete, create a new one by repeating the previous procedures [New population].
    a. Select two parent chromosomes from a population based on their fitness level; the higher the fitness level, the greater the likelihood of selection [Selection].
    b. With a crossover probability crossover the parents to form a new offspring (children) are formed. If no crossover was performed, offspring is an exact copy of parents [Crossover].
    c. New offspring at each locus (position in a chromosome) have a mutation chance [Mutation].
    d. Place new offspring in a new population [Accepting].
4. For a further algorithm run, use the newly created population [Replace].
5. Stop and return the best solution in current population if the final condition is satisfied. [Test].
6. Go to step 2 [Loop].

IV GENETIC OPERATORS

Genetic algorithms use three operators like selection, crossover, and mutation to find superior solutions.

1. Selection: The selection process involves choosing elite members of the population to be parents in order to produce kids. Fitness levels serve as a criterion for determining an individual's elite status. There exist numerous techniques for choosing the optimal set of chromosomes, including but not limited to roulette wheel selection, Boltzmann selection, rank selection and elitist selection. These techniques are briefly explained:

1.1 Roulette Wheel Selection: Fit parents are chosen for the wheel. Better chromosomes increase an individual's chances of being selected. Envision a roulette wheel containing all of the population's chromosomes, each of which is positioned large in accordance with its fitness role.

1.2 Rank Selection: When fitness diverges significantly, the preceding selection technique will not function properly. Each chromosome obtains fitness according to this classification after population is initially sorted by fitness via rank selection. The greatest will have fitness N (number of chromosomes in population), while the worst will have fitness 1, second worst 2, etc. Then, each chromosome gets a chance to be selected. Then chance of being selected is proportionate to place in sorted list. The best chromosomes are not much separates from the rest. Therefore this tactic could cause a delay in convergence.

1.3 Elitism Selection: The best chromosome is likely to be lost when we generate a new population. These best chromosomes are copied to a new population first in a process called elitism. The remainder is finished in a conventional way. GA performance can be swiftly enhanced by elitism.

2. Crossover: In GA, a set of operators certain members determines the generation of successors. Crossover and mutation are used most frequently. This operator takes 2 parent strings and produces two new offspring by copying certain parts of each parent string. The bit of each child at a given location is a copy of one of the bits of each parent at that same location. Which parent gives the bit for placement is determined by an additional string called the crossover mask. There are three types of crossover: uniform, 2-point, and 1-point.

2.1 1-Point crossover: Crossover mask is created by beginning with a string consisting of n consecutive 1s and terminating with the number of 0s. This produces children where one parent contributes the first n bits and the other parent contributes the remaining bits. The crossover mask is made after the crossover point n is randomly selected every time crossover operator is used.

2.2 2-Point Crossover: By replacing the middle of the second parent string with intermediate segments of one parent, 2-point crossover creates children. Stated differently, the crossover mask is composed of three strings: a string that starts with n0 zeros, a continuous string of nl ones, and the number of zeros required to finish the string. The randomly chosen integers n0 and nl provide a mask each time the two point crossover operator is applied. To create two children, the roles of the two parents are again reversed.

2.3 Uniform Crossover: Bits that are equally sampled from both parents are combined. In this case, the crossover mask is generated as a random bit string, where each bit is selected independently of the others at random.

3. Mutation: One kind of operator creates children from a single parent, while the other is a recombination operator that creates children by fusing the components of two parents. The bit string experiences small, random changes as a result of the mutation operator's selected, random modification of one bit's value.

V FITNESS FUNCTION

Criteria for rating potential hypotheses and probabilistically choosing which ones to include in population of following generation are defined by the fitness function. A component of the fitness function that rates the rule's classification accuracy across a set of given training instances is usually present if the task involves learning classification rules. Extra components are frequently added, such the regulation's intricacy or generality. Taking into account that discrete search space problems can be solved with the help of genetic algorithms. Because of this, GA is a very effective optimization tool in addition to being relatively simple to use. Each of the strings in GA, referred to as chromosomes, provides a potential solution to the current issue. Every chromosome aims to maximize its fitness value. A population is made up of two chromosomes and the fitness that goes along with them [9]. Populations created during GA iteration are referred to as generations. The nomenclature used in human and GA is summed up as indicated in Table 1, as stated in [10].

Table 1: Comparative Terminology

Human Genetic	GA Terminology
Chromosomes	Bit strings
Genes	Features
Allele	Feature value
Locus	Bit position
Genotype	Encoded string
Phenotype	Decoded genotype

Chromosomes, which are potential solutions, are assessed for fitness with a function known as the goal or fitness function. Put another way, the objective function yields numbers that are used to order the designated chromosomes in the population. Problem that has to be solved determines how fitness function is expressed. Fitness describes standards of grading for possible hypotheses and selects which ones to incorporate in the population in a probabilistic manner. A component of the fitness function that rates the rule's classification accuracy across a set of given training instances is usually present if the task involves learning classification rules. When interpreting the bit-string hypothesis as a complex procedure (e.g., when the bit string represents a collection of if-then rules that will be chained together to control a robotic device), the fitness function may measure the overall performance of the resulting procedure rather than the performance of individual rules.

VI CLASSIFICATIONS OF GENETIC ALGORITHMS

Meta-Heuristic is a process used to try and solve extremely difficult optimization and search issues. Meta-heuristics use search algorithms or low-level heuristics, in contrast to other heuristics. As a result, meta-heuristics employ more abstract concrete heuristics or algorithms. Retrieved solution depends on created collection of random variables. Meta-heuristics, in contrast to iterative techniques and optimization algorithms, doesn't guarantee a optimal solution for a given problem. Stochastic optimization is used by meta-heuristics, and the resultant solution depends on the collection of randomly generated variables.

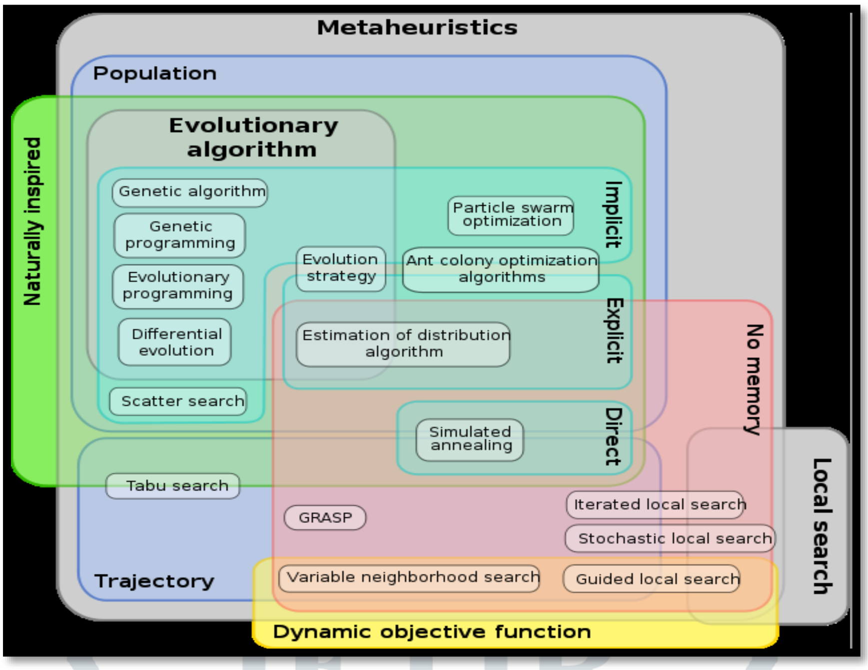

Figure 2: Meta-heuristics classification

Meta-heuristics are categorized by the following attributes:

- Meta-heuristics employ well-defined techniques to steer the search process. Meta-heuristic algorithms like GA are not random searches, even though they are randomized. They utilize historical data to focus the search on an area of the search space where performance is higher.
- Their goal is to locate nearly ideal solutions by effectively searching the state space. Thus, basic local search algorithms and intricate learning procedures are examples of meta-heuristic searches.
- Meta-heuristic algorithms are not problem-specific and are typically non-deterministic and approximate.
- Meta-heuristic algorithms are consequently applicable to a wide range of situations since they create assumptions for problem that needs to be solved.

GA is population-based, parallel searches with an element of learning. Stochastic crossover, mutation, and individual solution solutions are used in GA [11].

VII GENETIC ALGORITHM APPLICATIONS

Because of its adaptability, genetic algorithms have been used in a wide range of sectors and applications. Following is a discussion of a few of these applications:

1. Adaptable Hardware Programs

The manipulation of GA to make electronic is the foundation of this subject. Using stochastic operators, the genetic algorithm models automatically generate new configurations depending on the preexisting configurations. The model will eventually reach the desired configuration that the creator requires as it continues to evolve while operating within its environment [12]. A robot that can manipulate its built-in genetic algorithm to regenerate its configuration after a malfunction caused by environmental factors like electromagnetic waves, which might cause malfunctions in its usual configuration, is an example of a reconfigurable model.

2. Robotics

In order to create a complete and effective robot, designers and engineers must conduct experiments to determine all the necessary components for robot, including the matching hardware and matching software. To construct new robot that meets the new objectives, previously listed activities must be completed for each new mission. Many of these additional design constraints can be removed by tinkering with the genetic algorithm. The ability to automatically create a set of ideal designs that may be applied to particular tasks and activities will be made possible by genetic algorithms. This method can even be extended to create robots that can process more complex applications and carry out several duties. The process of navigating is another area where GA is used in robotics. GA offers navigation processes that are tuned so that the robot can arrive at its intended location without getting lost or colliding with surrounding objects. Every chromosome in the navigation algorithm denotes a set of nodes, which is a gene. Every gene contains two values: Boolean and x-y value that indicates current node can reach next node or not. With method, the robot will be able to choose the best path in addition to always finding a way to the destination without running into any obstacle.

3. Engineering design

The process of creating a new engineering design is difficult and laborious, but creating ideal design that consumes fewest resources possible while producing the greatest output is even more difficult. To do such a task flawlessly, a considerable deal of work and experience are needed. This is another instance where the genetic algorithm's functionality is put to use. Applications for computer-based engineering design can incorporate GA. When creating a new design for a particular problem, the application can examine many aspects of engineering design concepts by employing this technique. This method will help the designers find the weaknesses and potential points of failure in the design in addition to giving them the necessary design. Numerous technical sectors, including aerospace, civil, automotive, robotics, electronics, and mechatronics, are actively using this method. These represent a tiny portion of the domains in which genetic algorithms are now being applied to maximize improvement in results. Many other domains, including chemical analysis, financial strategies, marketing strategies, gaming, trip and cargo scheduling, telecommunication routing, and encryption and code cracking, also make use of GA. In conclusion, genetic algorithms are becoming more and more important in the technological and scientific domains of the present world.

4. Data Encryption

By utilizing crossover and mutation, genetic algorithms are being employed in the field of cryptology to create new, advanced encryptions. A cryptosystem is a collection of algorithms with secret keys that are used to encode data or messages into ciphertext and then decode them back to their original form. The model put forth by Shannon or a secret key system is depicted in the above figure. A novel symmetric block ciphering method known as ICIGA (Improved Cryptology Inspired by Genetic Algorithms) allows a random process to generate a unique session key. At the beginning of the ciphering process, the users are able to ascertain the block sizes and key lengths. Actually, the ICIGA was developed as a result of improvements made to the GIC system (Genetic algorithms Inspired Cryptography). The ICIGA divides the plaintext into equal-sized sections according to secret key. Initial step in the ciphering process involves splitting the material into equal-sized blocks, which are then utilized to generate the secret key. Other portions of the message will subsequently be deciphered using the secret key that is being developed.

5. Computer Gaming

When playing video games, a human player's opponent is frequently a sophisticated artificial intelligence (AI) system that uses genetic algorithms. In order to make sure the AI can learn from the past experience and get better; strategies that have been employed in the past are implemented utilizing GA. The AI can avoid making the same mistakes twice thanks to the learning approach, which makes the game more playable overall. The human player can now have a more realistic experience because they will occasionally need to adjust their strategy. Additionally, it helps to prevent the possibility that a human player would discover a series of actions that, in the end, result in success. To determine an instance's quality, it needs a fitness function and a way to express the problem in terms of the solution. An entity's altered instantiation is first given to the fitness function, which then assesses the entity's excellence. Then, this particular function is adjusted to fit the problem domain. Fitness can be simplified to time function in most circumstances, especially when code optimization is involved. The GA will instantiate the initial candidates as soon as a fitness function and genetic representation are created. To increase the candidates' fitness value, it will then employ the repeating range of operators, which consists of selection and selection.

VIII CONCLUSIONS

Greatest tool for applying fitness functions to a variety of common situations is the Genetic Algorithm (GA). In the field of artificial intelligence, genetic algorithms are a thorough method that can be used to locate the best answer in challenging search places. This is a heuristic method looks for optimal answer by utilizing past data. In contrast to combinatorial optimization issues, which have known sophisticated solutions, high complexity problems are a good fit for the genetic algorithm. The vast functionalities of genetic algorithms will be applied in practical scientific and industrial domains to solve a great deal of today's problems, and the development of agents capable of performing such tasks effectively and efficiently without human assistance has a bright future ahead of it. Furthermore, by generating numerous optimal solutions, the application optimizes problems. To find many solutions with the best accurate outcome, the majority of the operations in the fitness calculation process involve repeating and recalculating. It is recommended to implement methods such as Hash Table, Taguchi Method (TM), Binary Tree Algorithm, and Keep Last Generation Algorithm to achieve the most improvement in GA. These methods have been shown to greatly increase the performance of the simple GA approach.

REFRENCES

[1] Akbari, Z. 2011.A multilevel evolutionary algorithm for optimizing numerical functions. IJIEC (2). 419- 430.
[2] Noraini Mohd Razali. John Geraghty. 2011. A genetic algorithm performance with different selection Strategies. Proceedings of the World Congress on Engineering. Vol. 2.
[3] Shrivastava. Animesh and Singh Rajawat. 2014. An implementation of hybrid genetic algorithm for clustering based data for web recommendation system.. Int J Comput Sci Eng. Vol. 2(4): 6-11.
[4] Sangari, Fardin Esmaeeli, and Mehrdad Nabahat. 2014. Efficacy of Different Strategies in Graph Coloring with Parallel Genetic Algorithms. 138-141.
[5] Haldurai, L. and Vinodhini, V. 2015. Parallel Indexing on Color and Texture Feature Extraction using R-Tree for Content Based Image Retrieval. International Journal of Computer Sciences and Engineering. Volume-03. Issue-11. 11-15.

E-ISSN: 2347-2693.
[6] Sreedharamurthy, S K. and Sudarshana Reddy, H R. 2015. Feature Subset Selection Using Genetic Algorithms for Handwritten Kannada Alphabets Recognition. International Journal of Computer Sciences and Engineering. Volume-03. Issue-06. 94-99.
[7] Amin Dastanpour. Suhaimi Ibrahim and RezaMashinchi. 2016. Effect of Genetic Algorithm on Artificial Neural Network for Intrusion Detection System. International Journal of Computer Sciences and Engineering. Volume-04. Issue-10. 10-18.
[8] Tian J. Hu Q. Ma X. Ha M. 2012. An Improved KPCA/GA-SVM Classification Model for Plant Leaf Disease Recognition Journal of Computational Information Systems. 18. 7737-7745.
[9] Radwan A. Latef B. Ali A. and Sadek O. 2006. Using Genetic Algorithm to Improve Information Retrieval Systems. World Academy of Science and Engineering Technology. Vol 17(2). 6-13.
[10] Sivanandam S. N. Deepa S. N. 2008. Introduction to Genetic Algorithms Springer-Verlag. Berlin.Heidelberg.
[11] Taherdangkoo Mohammad. Paziresh Mahsa. Yazdi Mehran. Bagheri Mohammad. 2012. An efficient algorithm for function optimization: modified stem cells algorithm. Central European Journal of Engineering. Vol 3 (1): 3650.
[12] Zhang J. Chung H. Lo W. L. 2007. Clustering-Based Adaptive Crossover and Mutation Probabilities for Genetic Algorithms. IEEE Transactions on Evolutionary Computation. Vol 11(3): 326-335.