International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Navigating Evolution: An Overview of AI-Based Genetic Operators

Rahul Kumar[1], Sumit Dalal[2], Rohini Sharma[3], Sumiran[4]

[1]Student, ECE, Sat Kabir Institute of Technology and Management, Bahadurgarh, Haryana

[2] Assistant Professor, ECE, Sat Kabir Institute of Technology and Management, Bahadurgarh, Haryana

[3]Assistant Professor and Corresponding Author, GPGCW, Rohtak, Haryana

[4] Assistant Professor, ECE, Sat Kabir Institute of Technology and Management, Bahadurgarh, Haryana

ABSTRACT:

Natural phenomena is a common source of inspiration for researchers and engineers trying to solve difficult optimization challenges. A subclass of evolutionary algorithms known as genetic algorithms uses natural selection as a model to solve a wide range of issues. Genetic algorithms have advanced significantly with the introduction of artificial intelligence (AI), transforming a variety of industries, including aerospace and banking. This review explores the ideas, uses, and developments of artificial intelligence (AI)-based genetic algorithms, revealing their potential to influence problem-solving in the future. This paper examines the latest developments in genetic algorithms. The examination will focus on the genetic algorithms that the research community finds most interesting. The fresh and demanding researchers will benefit from this review, which will provide them a broader perspective on genetic algorithms. The implementation of well-known algorithms is described together with advantages and disadvantages. In order to help new researchers, the genetic operators and their applications are presented. There is coverage of the various study fields related to genetic algorithms. It is discussed where genetic operators, fitness functions, and hybrid algorithms might be researched in the future.

INTRODUCTION:

The idea of evolution is the foundation of genetic algorithms (GA). These algorithms begin with a population of feasible solutions represented as data strings called chromosomes. Genetic algorithms replicate the principles of natural selection by iteratively evolving and improving these solutions across generations through a process of selection, crossover, and mutation. Over time, fitness functions direct the algorithm towards optimal or nearly ideal answers by assessing the quality of each result. Genetic algorithms are incredibly versatile and have applications in a wide range of fields. In the world of finance, they maximize profits while reducing risks in investment portfolio optimization. They streamline production procedures in the manufacturing sector, cutting expenses and raising productivity. They support medication development and therapy optimization in the medical field, revolutionizing customized medicine. Genetic algorithms are widely used in robotics and telecommuting, providing creative answers to challenging issues.

The field has advanced to new heights thanks to the combination of AI and genetic algorithms. Genetic algorithm performance is improved by machine learning approaches, especially reinforcement learning and neural networks. Neural networks can evolve for particular tasks thanks to hybrid methodologies like neuro evolution, which combine the advantages of neural networks and genetic algorithms. In addition, genetic algorithms can now handle bigger and more complicated issues because to advancements in distributed systems and parallel computing, which speed up the optimization procedure. Figure 1 shows the complete processing of AI Based genetic algorithms.

Traditional GA

An optimization technique called the genetic algorithm (GA) is based on natural selection. The concept of the survival of the fittest theory is applied in this population-based search method [1]. By applying genetic operators on members of the population iteratively, new populations are created. The main components of GA are chromosomal representation, selection, crossover, mutation, and fitness function calculation. Random initialization is performed on a population (Y) with n chromosomes. Every chromosome in Y has its fitness calculated. A pair of chromosomes, designated as C1 and C2, are chosen from population Y based on their fitness value. To create an offspring, let's say O, single-point crossover operator with crossover probability (Cp) is fitted to C1 and C2. The generated offspring (O) with mutation probability (Mp) is then subjected to a uniform mutation operator, producing O′. The new progeny O′ is assigned to a new population. Until the new population matures, the present population will undergo selection, crossover, and mutation processes again. By utilizing crossover and mutation probabilities, GA dynamically modifies the search process until it reaches the best solution. The encoded genes can be changed by GA. GA is capable of evaluating numerous people and generating numerous ideal options. GA can so perform global searches more effectively. The excellent genetic architecture of the parent chromosomes are likely to be eliminated in the kids created by parent chromosome crossing. The Schema Theorem states that a modified schema must be used in place of the original template. The new schema preserves the original population from the early stages of evolution in order to preserve population variety. The proper schema will emerge at the conclusion of evolution to guard against any deterioration of superior genetic schema[2].

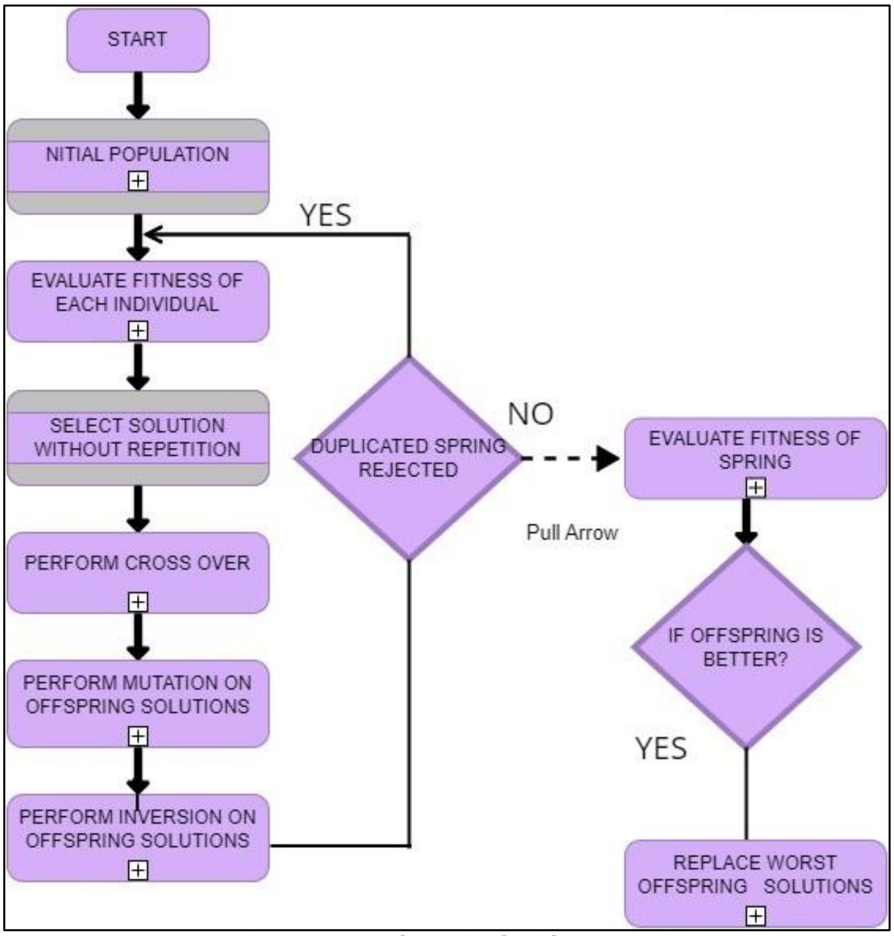
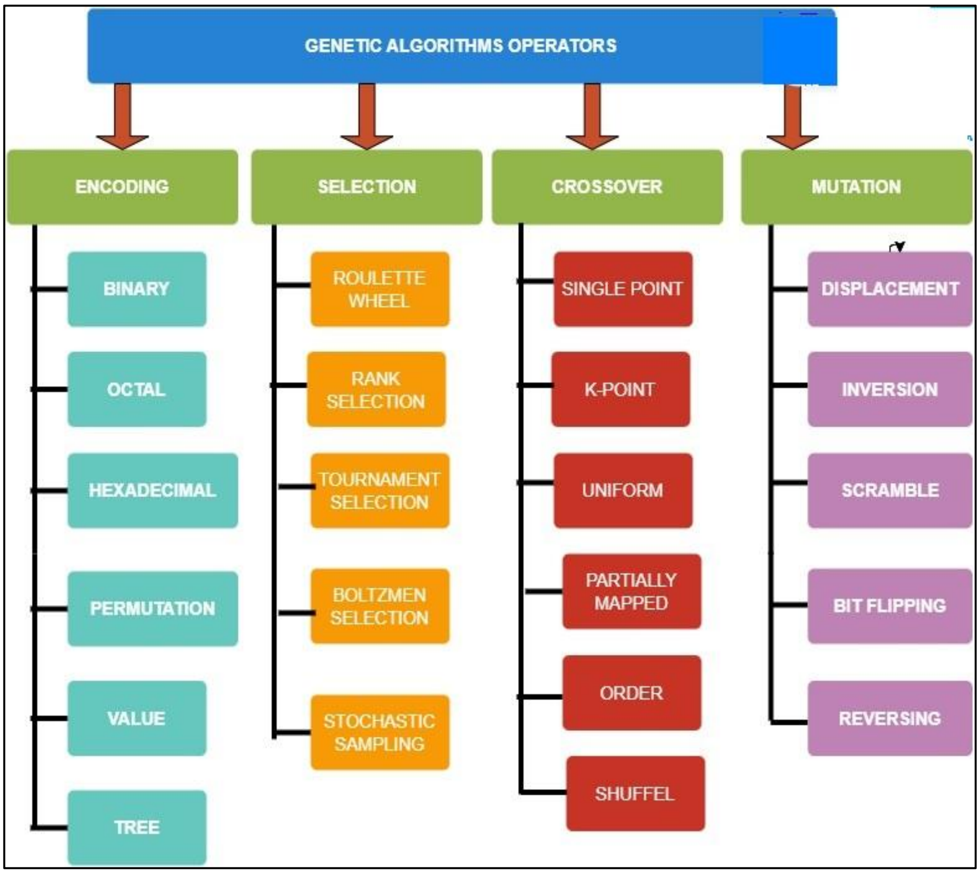


Figure 1: AI Based Genetic Algorithms Processing

DIFFERENT TYPES OF GENETIC OPERATORS

Various operators have been used by GAs in the selection process. Encoding schemes, crossover, mutation, and selection are examples of these operators (Figure 2).



Encoding Operators

The encoding method, which refers to the conversion of data into a certain form, is crucial for the majority of computer issues. It is necessary to encode the provided data in a specific bit string [3]. The problem domain determines how the encoding systems differ from one another. The encoding techniques that are well recognized include binary, octal, hexadecimal, permutation, value-based, and tree. The most popular encoding system is *binary encoding*. A string of 1 or 0 represents each gene or chromosome [4]. Every bit in binary encoding indicates a feature of the solution. It enables crossover and mutation operators to be implemented more quickly. But, the conversion to binary form necessitates additional work, and the algorithm's correctness depends on the binary conversion. The bit stream is modified in accordance with the issue. Because of epistasis and natural depiction, binary encoding schemes are inappropriate for many engineering design challenges. The gene or chromosome is indicated by an octal number (0–7) in the octal encoding method. The gene or chromosome is denoted by hexadecimal numbers in a hexadecimal encoding method (0–9, A–F) [5]. Usually, ordering problems make use of the permutation encoding approach. A string of integers designating a sequence's position serves as the encoding scheme's representation of a gene or chromosome. Genes and chromosomes are represented by strings of values in value encoding schemes. Real, integer, and character values are all possible for these. When using more complex values, this encoding approach can be useful in solving certain difficulties. Since such issues may cause binary encoding to fail. It is mostly used to determine the ideal weights in neural networks.

A tree of functions or commands is used in tree encoding to convey the gene or chromosome. You can use these commands and functions with any programming language. This is strikingly similar to how repression is represented in a tree structure [6]. Typically, this kind of encoding is employed in dynamic programs or expressions.

Table 1: A review on Encoding Schemes

Encoding Scheme	Benefits	Drawbacks
Binary	Simple to use and Quicker to complete	Absence of inversion operator compatibility
Octal	Simple to implement	Absence of inversion operator compatibility
Hexadecimal	Simple to implement	Absence of inversion operator compatibility
Permutation	Aid the inversion operator	Absence of inversion operator compatibility
Value	Value conversion is not necessary.	needs certain mutation and crossover
Tree	Operator is readily applicable	Tree design is challenging for some issues

Selection Techniques

In genetic algorithms, selection is a crucial stage that establishes whether or not a given string will take part in reproduction. The reproduction operator is another name for the selection stage. The selection pressure affects how quickly GA converges. The roulette wheel, rank, tournament, Boltzmann, and stochastic universal sampling are examples of popular selection methods. All potential strings are mapped onto a wheel in roulette wheel selection, and each string is given a specific piece of the wheel based on its fitness value. Following that, a random rotation of this wheel determines which particular solutions will be included in the creation of the following generation. Nevertheless, it has a number of issues, including mistakes brought up by its stochastic character. By adding the idea of determinism to the selection process, De Jong and Brindle altered the roulette wheel selection technique to eliminate errors. The modified version of roulette wheel selection is called rank selection. Instead of using fitness value, it makes use of the ranks. Each person is assigned a rank based on their fitness worth, giving them a chance to be chosen based on their ranks. The likelihood of the solution prematurely converging to a local minima is decreased by the rank selection approach.

Brindle first suggested the tournament selection method in 1983. The participants are chosen in pairs based on their fitness scores from a stochastic roulette wheel. Following selection, those with greater fitness values are added to the group of future generations [6]. Each person in this selection process is compared to all n-1 other people if they make it to the final population of solutions. Monte Carlo simulation uses sampling techniques and entropy as the foundation for Boltzmann selection. It assists in resolving the premature convergence issue [7]. It takes very little time to execute and has a very high probability of choosing the best string. Information loss is a possibility, though.

Table 2: A review on Selection Techniques

Selection Techniques	Benefits	Drawbacks
Roulette wheel	Simple to implement, Free from Bias	Threat of Early Convergence It is contingent upon the fitness function's variability.
Rank	Reserve diversity	gradual convergence, Sorting is necessary. Computationally costly
Tournament	Reserve diversity Parallel Execution	loss of diversity in a large-scale contest

	No sorting needed	
Boltzmann	Universal optimum attained	Computationally Affluent
Stochastic Universal Sampling	Fast Technique Free from Bias	Early convergence

Crossover operators

The genetic material of two or more parents is combined to create the offspring via crossover operators. The most well-known crossover operators include precedence preserving crossover, shuffle, reduced surrogate, uniform, partially matched, k-point, and cycle.

A crossing point is chosen at random in a single point crossover. Two parents whose genetic information is beyond that point will be switched with one another[8]. When two or more random crossover points are chosen in a two-point and k-point crossover, the parents' genetic information is switched in accordance with the segments that have been established. To create the new offspring, the parents' middle section is swapped out. Parent in a uniform crossover cannot be broken down into parts. It is possible to address each parent gene independently. We determine at random if we need to switch the gene for another chromosome at that place. The most popular crossover operator is partially matched crossover (PMX). This operator outperforms the majority of other crossover operators in terms of performance. D. Goldberg and R. Lingle proposed the partially matched (mapped) crossover [2]. For mating, two parents are selected. A child receives a portion of its genetic makeup from each parent in proportion to the genetic material donated by the first. Following this procedure, the absent alleles are replicated from the second parent.

In order to lessen the bias introduced by other crossover strategies, Eshelman et al. [9] devised shuffle crossover. In order to prevent bias from being introduced into the crossover process, the values of each individual solution are shuffled prior to the crossover and unshuffled following the crossover operation. Nevertheless, in recent years, this crossover has not been used very much. For solution representations, reduced surrogate crossover (RCX) minimizes needless crossovers when the parents share the same gene sequence. The foundation of RCX is the idea that GA, provided the parents have a sufficiently varied genetic makeup, creates superior offspring. Nevertheless, RCX is unable to generate superior offspring for parents with the same composition. Oliver [10] suggested cycle crossover. By referring to the positions of their parents, it aims to produce an offspring using parents in which each element occupy the position. It borrows certain components from the first parent in the first cycle. It uses the remainder of the components from the second parent in the subsequent cycle. A uniform crossover works well with big subsets. Compared to other crossover strategies, order and cycle crossovers offer superior investigation. A crossover that is partially matched facilitates greater exploration. Compared to other crossover strategies, the performance of partially matched crossover is superior. Premature convergence affects cycle crossings and reduced surrogates.

Table 3: A review on Crossover Techniques

Crossover Techniques	Benefits	Drawbacks
Single point	Simple Implementation	Fewer assorted solutions
Two and K-point	Simple Implementation	Fewer assorted solutions
Uniform	Unprejudiced investigation, scalable to big subgroups, enhanced recombination potential	Fewer assorted solutions
Order Crossover (OX)	Improved Exploration	Loss of data from the former individual
Cycle Crossover	Impartial Exploration	Precipitate convergence

Mutation Operators

The mechanism that preserves genetic diversity from one group to the next is mutation. The three most well-known mutation operators are scramble, simple inversion, and displacement. A substring of a given individual solution is displaced inside itself by the displacement mutation (DM) operator. To ensure that the final solution is legitimate and includes a random displacement mutation, a random location is selected from the provided substring for displacement. Exchange mutations and insertion mutations are two types of DM variations. A portion of a single solution is either swapped out for another portion or inserted in a different spot in exchange mutation and insertion mutation operators, correspondingly. In a single solution, the simple inversion mutation operator (SIM) flips the substring between any two given sites. The randomly chosen string is reversed and placed at a random spot using the SIM inversion operator. The scramble mutation (SM) operator randomly arranges the elements in a predetermined range of the individual solution and determines whether or not the freshly created solution's fitness value has increased.

Table 4: A review on Mutation Techniques

Mutation Techniques	Benefits	Drawbacks
Displacement Mutation	Simple Implementation	Possibility of Early convergence
Simple-Inversion Mutation	Simple Implementation	Possibility of Early convergence
Scramble Mutation	Imitates big number of genes	Worsening of solution quality in some complications

Genetic Algorithm in Artificial Intelligence

In artificial intelligence, the Genetic Algorithm concept stands out as a highly creative and successful strategy. This algorithm replicates the evolutionary process that produces answers to issues that would be too difficult for regular algorithms, drawing on concepts from genetics and natural selection. In genetic algorithms, the pieces that make up the population are referred to as genes in artificial intelligence. These genes, which are also known as chromosomes, create an individual within the population. All individuals are gathered into a search space that is generated. Every person in the search space has a code within a limited range. Every individual in the search space (population) receives a fitness score that indicates how well they can compete with one another. The genetic algorithm searches for and maintains each person according to their fitness score, giving the highest-scoring individuals the opportunity to procreate. In comparison to their parents, the new children are having better "partial solutions." Additionally, the search space is kept dynamic via genetic algorithms in order to accumulate fresh solutions (offspring). Until their offspring have no more characteristics or traits than their parents, this process continues (convergence). At the end, the population converges, leaving only the fittest solutions and their progeny—better solutions—to survive. New members of the population (offspring) have their fitness scores computed as well.

CONCLUSIONS:

Artificial intelligence (AI)-driven genetic algorithms offer a potent paradigm for resolving challenging optimization issues in a variety of fields. With ongoing developments in AI and computer power, genetic algorithms have the potential to completely transform problem-solving. Genetic algorithms offer novel approaches to the most difficult issues facing the modern world by combining the concepts of evolution with the power of artificial intelligence. Even with its advances, AI-based genetic algorithms still have drawbacks such the requirement for large amounts of computer power, premature convergence, and scaling problems. It will take cutting edge computational methods and hardware capabilities to overcome these obstacles. Future topics for research include investigating different representations and operators, integrating dynamic adaptation methods, and integrating multi-objective optimization. Extensive research has been done on the impact that genetic operators including crossover, mutation, and selection play in mitigating premature convergence.

REFERENCES:

[1] A. Abdelghany, K. Abdelghany, and F. Azadian, "Airline flight schedule planning under competition," *Comput. Oper. Res.*, vol. 87, pp. 20–39, 2017.

[2] D. E. Goldberg, "Ioci and the Traveling Salesman Problem," in *Proc., The First International Conference on Genetic Algorithms and Their Applications*, 1985, vol. 154.

[3] J.-Y. Lee, M.-S. Kim, C.-T. Kim, and J.-J. Lee, "Study on encoding schemes in compact genetic algorithm for the continuous numerical problems," in *SICE Annual Conference 2007*, 2007, pp. 2694–2699.

[4] S. G. Koolagudi, Y. V. S. Murthy, and S. P. Bhaskar, "Choice of a classifier, based on properties of a dataset: case study-speech emotion recognition," *Int. J. Speech Technol.*, vol. 21, no. 1, pp. 167–183, 2018.

[5] N. Krishnan, S. Muthukumar, S. Ravi, D. Shashikala, and P. Pasupathi, "Image restoration by using evolutionary technique to Denoise Gaussian and impulse noise," in *Mining Intelligence and Knowledge Exploration: First International Conference, MIKE 2013, Tamil Nadu, India, December 18-20, 2013. Proceedings*, 2013, pp. 299–309.

[6] K. Jebari and M. Madiafi, "Selection methods for genetic algorithms," *Int. J. Emerg. Sci.*, vol. 3, no. 4, pp. 333–344, 2013.

[7] C.-Y. Lee, "Entropy-Boltzmann selection in the genetic algorithms," *IEEE Trans. Syst. Man, Cybern. Part B*, vol. 33, no. 1, pp. 138–149, 2003.

[8] G. K. Soon, T. T. Guan, C. K. On, R. Alfred, and P. Anthony, "A comparison on the performance of crossover techniques in video game," in *2013 IEEE international conference on control system, computing and engineering*, 2013, pp. 493–498.

[9] F. J. Burkowski, "Shuffle crossover and mutual information," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, 1999, vol. 2, pp. 1574–1580.

[10] D. N. Mudaliar and N. K. Modi, "Unraveling travelling salesman problem by genetic algorithm using m-crossover operator," in *2013 International Conference on Signal Processing, Image Processing & Pattern Recognition*, 2013, pp. 127–130.