كلية الحاسبات والذكاء الإصطناعي
Faculty of Computers & Artificial Intelligence

Helwan University

Evolutionary Algorithm

DR.
Amr S. Ghoneim

Contents

Definition:

The optimization of functions is a crucial aspect of computational intelligence, as it aims to minimize objective functions in various domains. Our research concentrates on optimization and employs advanced techniques, including Genetic Algorithms (GAs) and Differential Evolution (DE), to achieve our goals.

We aim to identify the global minimum of benchmark optimization functions by continuously refining potential solutions. This approach enhances the efficiency of computational intelligence and elevates its accuracy in addressing complex problems.

Constrain Handling:

When optimizing a system, it's common to encounter constraints that restrict the range of potential solutions. However, several techniques can be employed to address these constraints. One effective technique is using penalty functions, which penalize the objective function for violating the constraints.

Another approach is to use repair techniques, which modify the potential solutions to ensure they comply with the constraints. Moreover, limiting the search space to feasible solutions alone can help ensure constraints are met.

Finally, decoder functions can also be useful as they map potential solutions to feasible solutions that satisfy all constraints.

Evolutionary Algorithm Component:

Genetic Algorithm (GA):

Genetic Algorithm is a heuristic search algorithm inspired by natural selection. It is a commonly used optimization technique that mimics the process of biological evolution. The algorithm works by iteratively improving a population of candidate solutions by applying genetic operators such as selection, mutation, and crossover.

The GA algorithm consists of a series of steps:

- First, everyone in the population is represented as a vector of real-valued numbers.
- Second, the fitness function evaluates the quality of an individual solution by computing the objective function value associated with the solution.
- Third, the population is a collection of individuals representing potential solutions to the optimization problem.
- Fourth, Roulette wheel selection is employed to select individuals for reproduction based on their fitness.
- Fifth, mutation and recombination operators are applied to introduce genetic diversity and exchange genetic information between parent solutions.
- Sixth, an elitist strategy is employed for survivor selection.
- Finally, the initial population is generated randomly within a predefined range of decision variable values, and the algorithm terminates after a fixed number of generations or when a specified termination criterion is met.

Differential Evolution (DE):

Differential Evolution is a stochastic population-based optimization algorithm that is commonly used for solving complex problems. It is a variant of the evolutionary algorithm that uses a different approach to generate new candidate solutions.

The DE algorithm works by iteratively improving a population of candidate solutions through the application of genetic operators such as mutation and recombination.

The DE algorithm shares some similarities with the GA algorithm.

- First, everyone is represented as a vector of real-valued numbers, like GA.
- Second, the fitness function evaluates the quality of an individual solution.
- Third, the population is a set of solution vectors comprising the DE population.
- Fourth, DE utilizes strategies such as best/1/bin or rand/2/bin to select parent vectors for mutation and recombination.
- Fifth, mutation and recombination operators are applied to generate new candidate solutions.
- Sixth, like GA, DE employs an elitist strategy for survivor selection.
- Finally, the initial population is generated randomly within a predefined range of decision variable values, and the algorithm terminates after a fixed number of generations or when a specified termination criterion is met.

Variation Operators:

Genetic Algorithm (GA):

In GA, two popular parent selection techniques are roulette wheel selection and tournament selection. Roulette wheel selection (that we use) selects individuals from the population with probabilities proportional to their fitness, while tournament selection randomly selects a subset of individuals and chooses the individual with the best fitness as a parent. Single-point crossover (that we use) and whole arithmetic recombination are two common recombination techniques in GA, while Gaussian (that we use) and uniform mutations are commonly used mutation techniques.

Differential Evolution (DE):

in DE, random selection and best selection are two popular parent selection techniques. Random selection selects three random individuals from the population for the mutation operation in each generation, while best selection chooses the individual with the best fitness along with two random individuals for the mutation operation. Binomial and exponential crossover are two common recombination techniques in DE, while differential and random mutations are commonly used mutation techniques.

These variation operators play a vital role in the performance of GA and DE algorithms. The selection of appropriate operator combinations can significantly improve the convergence rate and accuracy of the algorithm.

Population Management:

Genetic Algorithm (GA):

In GA, population management is achieved by employing a fitness-based survivor selection process. This process involves sorting individuals based on their fitness score, with the fittest individuals advancing to the next generation. This strategy ensures that only the best solutions are retained and propagated to the next generation, leading to a faster convergence towards the optimal solution.

Differential Evolution (DE):

In DE, a replacement strategy is employed which prioritizes individuals with lower fitness scores. This approach aids in promoting a widespread exploration of the search space, thereby preventing the algorithm from getting trapped in local optima. To further improve the overall fitness level, the differential mutation operator generates trial vectors that replace individuals in the population. Simultaneously, elite individuals, i.e., individuals with the highest fitness scores, are retained in the population to ensure convergence towards the optimal solution.

Population management is crucial for optimizing algorithms like GA and DE. The strategy used depends on the problem and search space characteristics. These algorithms are powerful tools for achieving optimal solutions in various applications.

Parameter Control and Tuning:

In optimization algorithms, such as Genetic Algorithm (GA) and Differential Evolution (DE), tuning parameters are crucial for achieving optimal performance. Extensive research has been conducted to control and tune parameters to enhance the efficiency of these algorithms. In this discussion, we will elaborate on approaches for controlling and tuning parameters of both GA and DE, based on the provided code.

Genetic Algorithm (GA) Approach:

Parameter Control:

- **Population Size:** The impact of population size on exploration and exploitation is significant. While larger populations enhance exploration, they also increase computational costs.
- **Mutation Rate and Crossover Rate:** These parameters control the balance between exploration and exploitation. Higher mutation rates promote exploration, whereas higher crossover rates promote exploitation.
- **Termination Criteria:** Vital to the success of the algorithm is defining criteria for terminating it, such as reaching a maximum number of generations or stagnation in fitness improvement.

Parameter Tuning:

- **Manual Tuning:** Experimenting with different parameter values based on domain knowledge and problem characteristics can be an effective way to tune GA parameters.
- **Parameter Optimization Techniques:** Alternatively, employing optimization techniques, such as grid search, random search, or metaheuristic optimization algorithms, to automatically tune parameters based on performance metrics can be advantageous.

Differential Evolution (DE) Approach:

Parameter Control:

- **Population Size:** Like GA, adjusting the population size affects exploration and exploitation in DE.
- **Mutation Factor and Crossover Rate:** These parameters control the exploration-exploitation trade-off. Higher mutation factors promote exploration, while higher crossover rates promote exploitation.
- **Termination Criteria:** Define criteria for terminating DE, like GA.

Parameter Tuning:

- **Manual Tuning:** Experiment with different parameter values based on problem characteristics and performance evaluation.
- **Parameter Optimization Techniques:** Employ optimization techniques to automatically tune parameters based on performance metrics.

Preserving Diversity Approach:

When working with optimization algorithms like Genetic Algorithm (GA) and Differential Evolution (DE), preserving diversity is crucial to prevent premature convergence and ensure thorough exploration of the search space. In this article, we will explore and implement an appropriate method for diversity preservation, complete with accompanying code.

Crowding:

Crowding is a technique used in evolutionary computation to maintain diversity. It penalizes individuals based on their proximity to others in the same population. This promotes diversity by discouraging clusters around local optima. Crowding distance has been proven to be a reliable way of preserving diversity in various applications.

Report of the result:

In our project, we started by selecting a random population that we would calculate fitness for it. Then we evolve by selecting, mutating, and recombining the parents to form better offspring with better fitness.

Differential Evolution (DE):

In DE, we start by setting N-dimensions of 2 to form an individual with an array with 2 variables [X, Y], and for the 3rd dimension (Z axis) we represent fitness. Then, we set a population size of 50 to form 50 arrays of a single individual, forming 50 coordinates in the 2-dimensional plane. After that, we set the mutation factor as 0.8 to not stray far away from the search space and a crossover rate of 0.9 to ensure the exploration of the algorithm (searching the state space with minor restrictions). Finally, We set the max generation to 100 due to computational abilities (we could increase the max generation to an infinite number depending on the benchmark function and the minimum accuracy of the model required).

Genetic Algorithm (GA):

In GA, we start by setting N-dimensions of 2 to form an individual with an array with 2 variables [X, Y], and for the 3rd dimension (Z axis) we represent fitness. Then, we set a population size of 50 to form 50 arrays of a single individual, forming 50 coordinates in the 2-dimensional plane. After that, we set the mutation rate as 0.1 to not stray far away from the search space (but add a little bit of diversity) and a crossover rate of 0.7 to exploit useful genetic traits. Finally, we set the max generation to 100 due to computational abilities (we could increase the max generation to an infinite number depending on the benchmark function and the minimum accuracy of the model required).

Conclusion:

In the end, the optimization of functions is an essential aspect of computational intelligence, aimed at minimizing objective functions in various domains. Genetic Algorithms (GA) and Differential Evolution (DE) are advanced techniques employed to achieve optimal solutions. While optimizing a system, constraints may restrict the range of potential solutions. However, several techniques such as penalty functions, repair techniques, limiting the search space, and decoder functions can be employed to address these constraints. The GA and DE algorithms have unique features and variations, including selection, recombination, and mutation techniques, which can be applied based on the optimization problem's nature and complexity. In summary, the GA and DE algorithms are effective optimization techniques that can enhance the efficiency and accuracy of computational intelligence in addressing complex problems.