Jordanian Journal of Computers and Information Technology (JJCIT), Vol. 10, No. 03, September 2024.

# CSGA: A DUAL POPULATION GENETIC ALGORITHM BASED ON MEXICAN CAVEFISH GENETIC DIVERSITY

Esra'a Alkafaween[1], Ahmad Hassanat[1], Ehab Essa[2] and Samir Elmougy[2]

*(Received: 11-Feb.-2024, Revised: 5-Apr.-2024, Accepted: 27-Apr.-2024)*

## *ABSTRACT*

*Genetic algorithms (GAs) are search algorithms based on population genetics and natural-selection concepts. Maintaining population variety in GAs is critical for ensuring global exploration and mitigating the risks of premature convergence. Rapid convergence to local optima is one such challenge in the application of genetic algorithms. To address this issue, we provide Cave-Surface GA (CSGA), an alternative method based on the Dual Population GA and inspired by the genetic variety observed in Mexican cavefish. Through inter-population cross-breeding, CSGA increases diversity via a secondary population (cave population) and facilitates the exchange of information between populations, effectively counteracting premature convergence. Several experiments are carried out utilizing benchmark instances of the Traveling Salesman Problem (TSP) obtained from TSPLIB, a well-known TSP problem library. Our experimental results over many TSP instances show that CSGA outperforms both classic GAs and other GAs that use diversity-preservation techniques, such as Multipopulation GA (MPGA). CSGA has the potential to give promising solutions to challenging optimization issues like TSP.*

## *KEYWORDS*

*Cave-surface GA, Diversity, GAs, Premature convergence.*

## 1. INTRODUCTION

GAs are one of the most well-known types of evolutionary algorithms [44]. The GA is based on the principles of biological evolution, which were first devised by John Holland [25] at the University of Michigan in the 1970s [19]. GA was created to investigate processes in natural systems and to construct artificial systems that preserve the adaptability and resilience of natural systems [18], [37].

The GA is regarded as an optimization technique, since it has demonstrated its durability and effectiveness in solving many challenges, such as: image recognition, combinatorial optimization, machine learning, computer networks, neural networks, …etc. [29]. Many combinatorial optimization problems in engineering and sciences have been effectively handled using GAs. There are many recent examples of the use of GA for combinatorial optimization. Examples include: TSP [30],[7] where the TSP is widely considered as a standard testbed of numerous combinatorial optimization strategies [8], routing problems [10], location problems [51] and scheduling problems [47][16].

The GA employs a set of solutions represented by a unique encoding. During the GA-implementation process, each solution or individual is assigned a fitness value that serves as a measure of the GA's performance. Each individual's fitness is directly related to the objective function of the optimization issue under consideration. The present individual population can be adjusted to form a new population utilizing three operations described by Holland: selection, crossover and mutation operators [38], [40],[22] and [5]. The selection operator selects which chromosomes in the population are permitted to reproduce, and generally, the chromosomes with the highest fitness are picked to generate more children than the others [6]. Sub-parts of two chosen chromosomes are exchanged by crossover operators [21],[24]. On the other hand, mutation operators randomly alter the allele values at certain chromosomal regions [23], [4] and [27]. The Ątness of the latest recent parent generation serves as an iterative guide for the searches in GAs. Every time we use GAs to solve an optimization problem, thousands of unique solutions are generated and to create offspring, the resulting solutions are assessed and recombined [27].

It is critical to provide a diversified population in order to achieve the best overall solution and

1. E. Alkafaween and A. Hassanat are with Computer Science Department, Mutah University, Al-Karak, Jordan. Emails: esra_ok@mutah.edu.jo and hasanat@mutah.edu.jo
2. E. Essa and S. Elmougy are with Department of Computer Science, Mansoura University, Mansoura, Egypt. Emails: ehab_essa@mans.edu.eg and mougy@mans.edu.eg

extensively explore the search space. According to the study of Osuna et al. [34], the amount of population diversity is a crucial factor contributing to premature convergence. In a GA, premature convergence occurs when a few highly ranked individuals dominate the population, forcing it to converge to a local optimum rather than the global optimum. According to the studies [33, 36], the main cause of premature convergence is a decline in population variety. This happens when the GAŠs population reaches a point where the genetic operators are unable to produce offspring who outperform their parents. It is crucial to protect population diversity throughout evolution in a GA in order to prevent premature convergence [32] and [31]. for maintaining diversity in GAs and avoiding the risk of premature convergence, numerous previous works used various methods, which include [35]: improvement of the genetic operators (mutation, crossover and selection) ( [48], [6]-[4]), dynamic parameter control [42], crowding method [12], MPGAs [45], a multi-objective evolutionary algorithm, dual population GA (DPGA) [36], primal-dual GA [49], …etc.

This research provides additional significant contributions, mostly to the area of enhanced GA performance using a new approach in the form of a dual-population GA inspired by the genetic variety discovered in Mexican cavefish, fittingly termed the 'Cave-Surface Genetic Algorithm' (CSGA). Due to its use of an additional population, CSGA falls under the category of MPGA. The study's specific contributions are:

- Encouraging diversity of the GA: Permitting the insertion of important and contextually relevant features into an individual's chromosome, CSGA allows the natural introduction of genetic variation.
- Mitigation of premature convergence: By keeping a diversified population throughout the evolutionary process, CSGAŠs novel design seeks to prevent early convergence and allow for the exploration of a larger solution space.
- Enhancement of solution quality: By virtue of its distinct characteristics and structures, CSGA shows gains in the quality of solutions produced, advancing the performance of genetic algorithms. These contributions reflect substantial advances in the field, providing useful insights and prospective applications for both practitioners and researchers.

Our methodology will be tested by applying it to instances of the Traveling Salesman Problem (TSP) supplied by the TSPLIB, a well-known resource of TSP problems [39]. To evaluate its performance and effectiveness, we will conduct a comparison analysis, pitting CSGA against the standard GA, MPGA as well as Particle Swarm Optimization (PSO).

The remaining part of this paper is structured as follows: Section 2 goes over similar works. The proposed algorithm is shown in Section 3. Section 4 presents experimental data to demonstrate the efficacy of the proposed approach. Section 5 concludes with a conclusion and discussion of future work.

## 2. RELATED WORK

Many approaches have emerged in recent years to enhance and sustain population diversity and thus reduce premature convergence. This helps improvement by giving global exploration assistance and getting access to different global and local optima [18].

Du et al. [14] suggested the use of elitism and distance to reduce genetic drift. Elites remain in place. Candidates for selection who are farthest from each elite are also retained to preserve diversification. In their studies, three EAs are used, including a GA, which is called every generation to maintain diversity. The second algorithm, DE/rand/2/bin, is a fundamental Differential Evolution (DE) algorithm. The third EA uses CoBiDE, a cutting-edge DE algorithm.

In [11], Osuna gave a great number of robust experimental and theoretical studies for EA to show how and why diversity plays a crucial role. Different diversity methods have been compared in a number of test functions within the framework of various EAs. The results obtained from the study shed light on how factors and mechanisms related to apparent diversity influence the search behavior of evolutionary algorithms, both in the presence and absence of diversity. These studies particularly point out which diversity strategies work for particular issues and which don't. Most significantly, they describe how to create the best evolutionary algorithms for the issues at hand.

To address the problems of exploration and exploitation, an enhanced GA-based new selection strategy, stairwise selection (SWS), was introduced. Its overall performance was compared to those of many other selection methods by employing 10 well-known benchmark functions across multiple dimensions. Furthermore, the study compared the statistical significance of the proposed SWS. The empirical results, supported by the graphical representation, showed that the SWS outperformed other competing systems in terms of stability, efficiency and durability, as evidenced by the authentication of a performance index [20].

Hassanat et al. [21] proposed two innovative deterministic crossover and mutation rate-control strategies: Dynamic Decreasing of High Mutation/Dynamic Increasing of Low Crossover (DHM/ILC) and Dynamic Increasing of Low Mutation/Dynamic Decreasing of High Crossover (ILM/DHC). These methods are dynamic, allowing for linear changes in both crossover and mutation operator rates as the search advances. Experiments on 10 instances of the Traveling Salesman Problem (TSP) were carried out to assess the efficiency of the suggested techniques. These experiments' results confirmed the efficacy of the proposed techniques.

Shojaedini et al. [41] used an adaptable genetic operator to choose high-fitness individuals as parents while mutating low-fitness ones. During the mutation phase, a training technique was used to gradually learn which gene is the best replacement for the mutant gene. By learning about genes, the suggested technique adaptively balances exploration and exploitation. The algorithm uses this information to enhance the final outcomes during the last iterations.

Hussain and Muhammad [26] presented a new split-ranked selection operator that provided a solid trade-off between exploration and exploitation. The proposed solution solves the fitness-scaling problem by ranking individuals from poorest to strongest depending on the calculated fitness scores. A series of experiments was carried out of some conventional operators and simulation studies using TSPLIB instances.

Inspired by the theory of natural selection, Albadr et al. [3] proposed a novel GA based on natural selection (GABONST), to better control over exploitation and exploration in optimization problems. According to the study, GABONST has outperformed the regular GAs in fifteen different standard test objective functions based on implementation and results. The algorithm's efficacy is ascribed to its capacity to focus on the more promising portions of the search space, which is accomplished by a well-balanced combination of exploration and exploitation.

Koohestani [28] proposed a permutation-based GA for tackling combinatorial optimization issues to improve the effectiveness of permutation-based GAs and to aid in developing high-quality solutions. A new edition of the so-called Partially Mapped Crossover is the main component of this GA. To evaluate the usefulness and efficiency of this crossover operator, two sets of experiments were carried out on popular benchmark problems.

The Population Diversity Controller-GA (PDC-GA) technique was devised as a distinctive feature-selection approach to reduce the search space during the construction of a machine-learning classifier. To effectively manage population diversity during the exploration phase, the PDC-GA combines GA with k-means clustering. When approximately 90% of the solutions become concentrated in a single cluster, an injection approach is employed to redistribute the population, ensuring a controlled level of diversity within the population [2].

A multi-objective binary GA with an adaptive operator-selection mechanism (MOBGA-AOS) was proposed by [48]. MOBGA-AOS employs five crossover operators, each with a unique set of search criteria. Each of them is allocated a probability based on how they perform during the evolution process. The proposed approach was compared against five well-known evolutionary multi-objective algorithms using ten datasets. MOBGA-AOS can remove a significant number of attributes while keeping a low classification error, according to the experimental results. Furthermore, it can handle high-dimensional feature-selection applications.

To solve the difficulties of readily slipping into a local optimum, low solution quality and sluggish convergence speed when solving TSP using GA, a GA incorporating jumping gene and heuristic operators (GA-JGHO) was presented by [50]. This algorithm features several improvements: a bidirectional heuristic crossover operator, enhanced roulette selection, a combination mutation operator and a jumping gene operator to prevent the formation of many similar individuals in the population. To avoid the development of nimiety identical to those within the population, a unique operator was included. In addition, the local search operator was added to boost exploitation potential.

According to the preceding analysis of the literature, different approaches and algorithms have been proposed to handle distinct issues in their respective sectors. While each of these approaches has made substantial contributions and breakthroughs, it is crucial to remember that none of them is perfect and there is still much opportunity for development. The wide range of proposed algorithms and methodologies emphasizes the importance of the ongoing study and development in this sector. Each method has advantages and disadvantages and it is critical to identify areas where improvements can be made to improve their performances. Subsequently, while the literature study highlights the availability of methods and algorithms, it also underscores the importance of further breakthroughs and improvements. Researchers can help build more effective and more efficient algorithms in the future by addressing the inadequacies of existing approaches.

## 3. METHODS

Many new concepts and ideas were incorporated into GAs. Following these concepts, we propose a dual population for GAs inspired by cavefish. Cave-dwelling species have offered scientists valuable knowledge regarding the evolutionary modifications of traits in response to distinct environmental and ecological limitations, as highlighted since Darwin's publication of "Origin of Species" [43]. Among such species, the Mexican blind cavefish stands out as a powerful research model due to its well-documented evolutionary lineage, clear ecological context and the presence of independently evolved cave populations. This species provides researchers with an excellent opportunity to investigate the factors contributing to convergent evolution [17]. Many of the cave-derived characteristics of cavefish, such as eye loss, loss of schooling and sleep loss have evolved repeatedly through independent origins and frequently by using various genetic pathways across caves [9] and [15], see Figures 1 and 2. This recurring evolution is a powerful feature of the Mexican blind cavefish system.

Figure 1. Surface fish. Adapted from (Bradic, Martina et al., 2012, with permission) [9].

Figure 2. Cave fish. Adapted from (Bradic, Martina et al., 2012, with permission) [9].

This work is based on the study carried out by Bradic et al. [9] on cavefish, which concluded the following points:

1) Many cavefish often get migrant fish from the surface and researchers continue to note that as one descends further into the cave, the frequency of surface fish in the pools rises.
2) Many fish in caves often accept migratory fish from the surface.
3) Estimates of migration rates and population sizes verified the concept that the influx of genes from surface populations and their effective population sizes are linked to the genetic diversity of cave populations.
4) Several of the cave populations were distinct and had increased genetic diversity, which was associated with rather high levels of migration from the surface. There was a significant gene flow in both ways between surface and cave populations.

Based on these conclusions, we proposed a mechanism for the GA, called: Cave-Surface GA (CSGA), in the hope of producing good individuals, increasing the diversity of the population and thus improving the efficiency of the GA. This mechanism uses two distinct groups of populations: The primary population is called the Cave, while the secondary population is called the Surface. The Cave population plays the same role as the GA, while the Surface population only provides diversity to the Cave population through cross-breeding.

CSGA begins with two populations that are generated at random, the Cave population and the Surface population. Individuals in each population are assessed using the same fitness functions. The Cave population undergoes evolutionary changes through a combination of inbreeding within the same population and cross-breeding with individuals from Surface populations. On the other hand, the surface population evolves primarily through inbreeding between parents from the same population. Next, we will describe the fitness function that was utilized for both populations. Subsequently, we will explain the evolutionary mechanism, including the processes of reproduction and survival selection, for both populations.

A) **Fitness function**: The population fitness function is just the objective function of the specific problem. In CSGA, the two populations use the same fitness function.
B) **Evolutionary procedure**: Traditional MPGAs reproduce simply by inbreeding among parents belonging to the same population. Their populations interact by exchanging certain individuals in accordance with a predetermined policy. Because MPGA populations share the same evolutionary goals and fitness function, excellent migrants are quickly absorbed into the new population. However, since the CSGA populations have similar fitness functions, there is no migration process and we will use cross-breeding as a process to enrich diversity in the populations.

The CCSGA generates offspring through both inbreeding and cross-breeding processes to facilitate the exchange of information between populations. Cross-breeding takes place when an individual from the Cave population reproduces with an individual from the Surface population, resulting in offspring that possess genetic material from both populations. These cross-bred offspring often exhibit fitness values that enable their survival in either population due to the combination of advantageous traits from both sources.

A standard GA chooses two parent chromosomes for a cross-over operation and produces two offspring. CSGA, on the other hand, has two extra parameters than GAs: the cross-breeding interval (CI) and the cross-breeding rate (CR). The cross-breeding interval (CI) is the number of generations between each cross-breeding and cross-breeding rate (CR) is the number of individuals selected from each population at the time of cross-breeding. These factors have an impact on the accuracy of the results, as well as on the computation time.

Based on the crossbreeding rate, CSGA randomly selects a number of parents from the two populations for recombination and generates two offspring through each cross-over operator between two parents. Then, through selection for local survival, one resulting offspring is selected to be a member of the next generation of Cave population, whereas the other offspring is sent to the Surface population, as shown in Figure 3. This procedure is repeated for each of the two candidate parents and in addition, this process does not take place in every generation, but takes place through specific generations, based on CR rate. The CSGA psudocode is shown in Algorithm1.

**Algorithm 1: CSGA Algorithm**

**Begin**

**step1:** initialize input parameters of problems: crossover rate, mutation rate, crossbreeding rate (cr), crossbreeding interval (ci), max generation.
**step2:** initialize two subpopulations, cave population (cp) and surface population (sp).
**step3:** For each subpopulation, repeat the following steps until the termination criterion is met.
**step4:** Calculate fitness value;
**step5:** inbreeding:
a. Selection
b. Crossover
c. Mutation
**step6:** crossbreeding (based on ci and cr do):
a. choose individual from cp and choose individual from sp.
b. Crossover
c. Move offspring one to
cp and the second to sp.
**step8:** output the final best solution.

**End.**

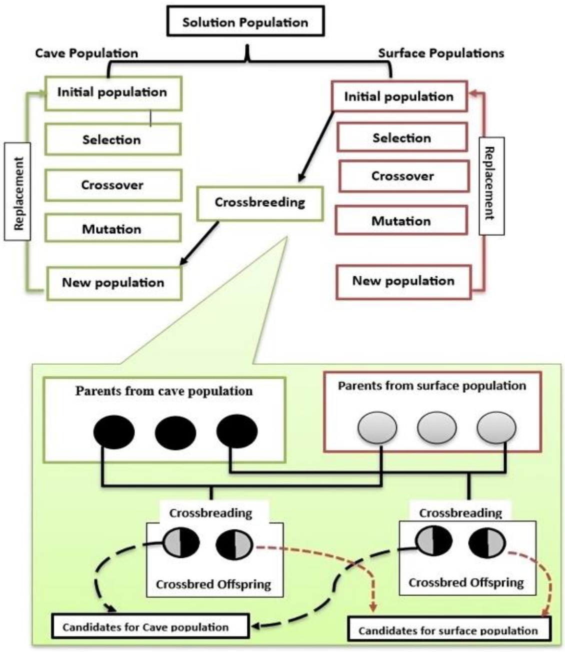


Figure 3. CSGA method; candidate offspring for cave and surface populations.

## 3.1 CSGA Procedure

The method begins by initializing the Cave population and Surface population, as well as the cross-over rate, mutation rate, cross-breeding rate, cross-breeding interval and maximum generation. The size of both populations is the same. Initially, the Cave population goes through the traditional GA evolution cycle. In Step 2, the fitness of the cave and surface populations is calculated using an objective function. Step 3: Inbreed both the Cave and Surface populations. The proper number of parents is then determined for reproduction based on the cross-over rate. The Cave population and the Surface population are then provided with diversity *via* mutation.

Cross-breeding between separate populations is performed in Step 4. Cross-breeding on the Cave and Surface populations produces a particular number of offspring. In cross-breeding, the intermediate Cave population, the intermediate Surface population and their offspring form a candidate set for the Cave and Surface populations' next generation. Step 5 evaluates created candidate sets using a fitness function for both populations and CSGA develops until either the maximum number of generation sets is attained or the algorithm provides the best possible solution to the present problem.

Table 1. TSP benchmark dataset.

| Class | Instance size | TSP instances |
|---|---|---|
| 1 | size < 200 | a280, att48, berlin52, bier127, ch130, ch150, eil51, kroA100, lin105, pr76, pr144 |
| 2 | size ≥ 200 | Lin318, ali535, rat783, kroB200 |

## 4. EXPERIMENTAL SETTINGS, RESULTS AND DISCUSSION

To verify the performance of the proposed algorithm, we conducted two sets of experiments on different TSP instances, which are given by the TSPLIB [39], which contains between 40 and 800 vertices. It is crucial to note that the TSP is used in this study only to compare the proposed CSGA to other methods in terms of diversity, rather than to seek a superior solution to the TSP problem. Simulation experiments are performed in the Microsoft Visual Studio 2022 environment and the system's hardware and software specifications are as follows:

- 11th Gen Intel(R) Core (TM) i7-1165G7 @ 80GHz 2.80 GHz
- 8.00 GB of RAM
- Windows 11 Pro, 64-bit operating system.

In the conducted experiments, each algorithm was applied 10 times to multiple instances of the TSP. The results obtained from each execution were averaged to provide a comprehensive assessment. Our GA utilized the reinsertion strategy, specifically the expansion sampling method introduced by Dong et al. [13]. This strategy involves selecting only the best half of individuals, including both new individuals and individuals from the previous generation, to form the population for the next generation. In other words, during the production of a new generation, the old generation competes with the newly generated individuals and only the fittest individuals are retained.

### 4.1 First Set of Experiments

The proposed method is compared to established GA algorithms utilizing fifteen TSP instances and varied numbers of vertices. Table 1 shows how the selected TSP instances were divided into two classes based on TSP size. Table 2 displays the selected GA parameters used in our experimental setup. The results of the proposed algorithms evaluated on TSP instances are summarized in Table 3.

As illustrated in Table 3, the CSGA performed better than the GAs in 7 out of 11 instances, in the first class. As for the second class, CSGA achieved the best performance over the GAs in the four TSP instances belonging to that category. Furthermore, when we look at the table, we can see in the Min column that the proposed CSCA had the lowest cost in 12 of the 15 TSP instances. It is important to mention that the simple GA-algorithm parameters were utilized as shown in Table 2; we did not use any sophisticated parameter control procedures, since the main purpose of this paper is to evaluate the efficacy and to demonstrate the goodness of the proposed method compared to the GAs in terms of diversity, regardless of the parameters employed, neutralizing parameters' tuning effect.

Table 2. The selected GA parameters used in our experimental setup.

| Parameter | Value |
|---|---|
| Population size | 200 |
| Generation limit | 3000 |
| Initialization method | Random |
| Cross-over | One-point modified |
| Cross-over rate | 0.85 |
| Mutation | Exchange |
| Mutation rate | 0.08 |
| Selection | Truncation selection |
| CR | 5 |
| CI | 7 |
| Termination criteria | Generation limit |

Figure 4 depicts each algorithm's convergence to the shortest route. Again, CSGA outperforms GA on a280 in terms of convergence to the minimal value, indicating that the population diversity provided by the CSGA allows for better convergence. This is also evident in Figures 5 and 6 that show the average convergence of the GA and CSGA, in small and large TSP instances, respectively.

Table 3. The results achieved by the GA and CSGA algorithms for TSP instances after 3000 iterations.

| Class No. | Optimal Solution | Instance | GA | | CSGA | |
|---|---|---|---|---|---|---|
| | | | Min. | Average | Min. | Average |
| Class1 | 2579 | a280 | 6952 | 7587.1 | 5914 | 6541.2 |
| | 10628 | att48 | 35843 | 41766.4 | 35704 | 40468.3 |
| | 7542 | berlin52 | 8253 | 9123.1 | 8497 | 9572.5 |
| | 118282 | bier127 | 152453 | 170944.7 | 146855 | 161837.4 |
| | 6110 | ch130 | 8865 | 10000.7 | 8768 | 9777.2 |
| | 6528 | ch150 | 10114 | 10914.66667 | 9965 | 10906 |
| | 426 | eil51 | 465 | 478.3 | 476 | 502.5 |
| | 21282 | kroA100 | 27555 | 32230.6 | 27175 | 31655.4 |
| | 14379 | lin105 | 20153 | 24129.1 | 20006 | 23199.1 |
| | 108159 | pr76 | 134438 | 133195.8182 | 130101 | 143111.4 |
| | 58537 | pr144 | 112926 | 119005.8 | 117911 | 134050.4 |
| Class2 | 42029 | lin318 | 125686 | 139947.1 | 112936 | 119163.4 |
| | 202339 | ali535 | 9484 | 10249.5 | 8418 | 8827.6 |
| | 8806 | rat783 | 51871 | 53879.1 | 46348 | 47547 |
| | 29437 | kroB200 | 58704 | 67404.5 | 57500 | 61319.4 |

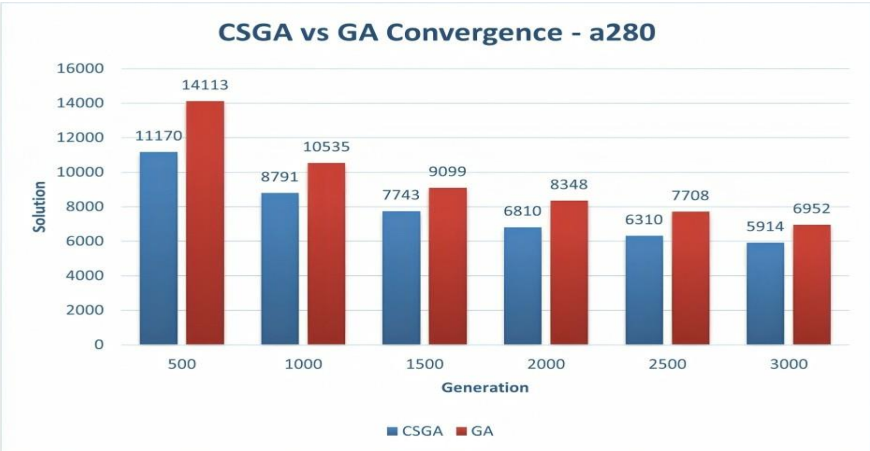


Figure 4. Average convergence of GA and CSGA for a280.

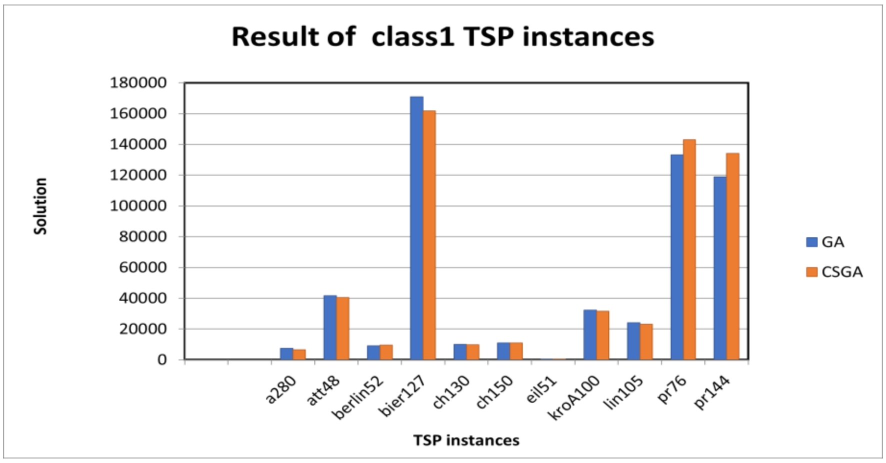


Figure 5. Average convergence of GA and CSGA for small instances from TSP (class1).

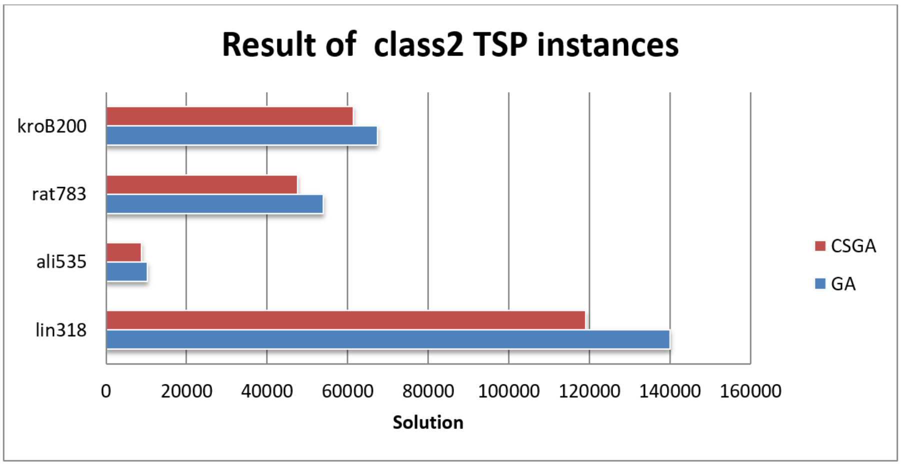


Figure 6. Average convergence of GA and CSGA for big instances from TSP (class2).

Figure 7 and Figure 8 show the route of two cities from TSBLIB, which are lin318 and kroA100, respectively. The resulting solution when applying the GA for the lin318 city is 124600 and the result when applying CSGA is 105240. As for the city of kroA100, the result of applying the GA was 32901 and as for the CSGA method, the result for this city was 26293. Note that these solutions and figures are the result of applying the two methods (GA and CSGA) to the same parameters found in Table 2.

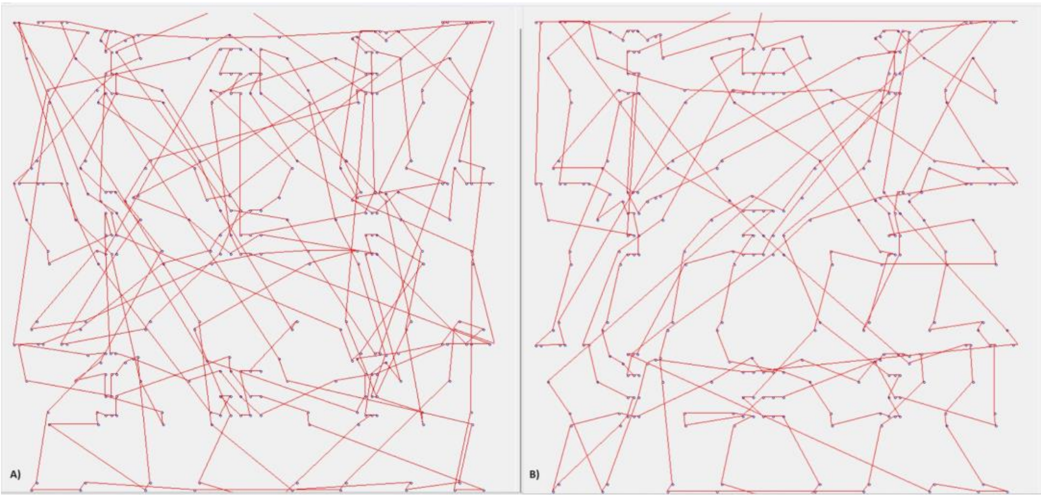


Figure 7. The resultant routes after applying the GA and CSGA methods on lin318 instance; (a) GA, (b) CSGA.

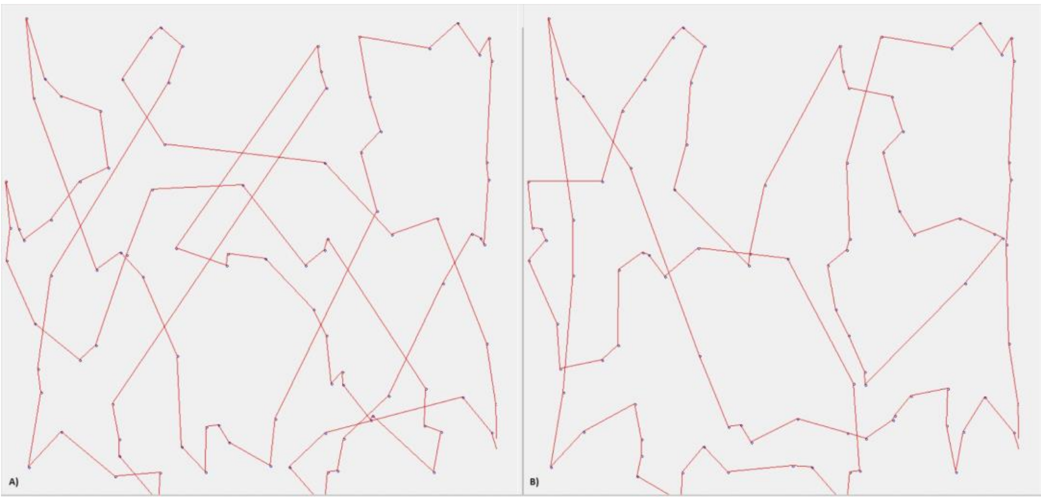


Figure 8. The resultant routes after applying the GA and CSGA methods on kroA100 instance; (a) GA, (b) CSGA.

## 4.2 Second Set of Experiments

The second group of experiments aims to study the effects of cross-breeding rate and interval on the proposed algorithm. It also aims to compare the proposed algorithm with one of the most famous methods that help diversify the population, which is MPGA. The results were also compared with those of a well- known optimization method; namely, PSO. We choose PSO, because GA and PSO are heuristic-based optimization methods and have many similarities in their inherent parallel characteristics [1]. In other words, PSO shares many similarities with evolutionary computation techniques, such as GA.

The parameters for the PSO were as follows: number of population =200, number of iterations =3000, cognitive component =1.5, social component=1.5 and inertia weight=0.7.

Table 4. Performance comparison of the CSGA, GA, PSO and MPGA.

| Optimal | Instance | GA | CSGA | MPGA | PSO |
|---|---|---|---|---|---|
| 2579 | a280 | 7587.1 | 6496.2 | 7306.3 | 30814.28302 |
| 10628 | att48 | 41766 | 39417 | 41766 | 124827.769 |
| 7542 | berlin52 | 9123.1 | 9411.7 | 8955.3 | 26070.21051 |
| 118282 | bier127 | 170945 | 165368 | 173151 | 581653.293 |
| 6110 | ch130 | 10001 | 9312.8 | 9354.6 | 41792.24209 |
| 6528 | ch150 | 10915 | 10650 | 10859 | 50091.11352 |
| 426 | eil51 | 478.3 | 497.9 | 476.2 | 1367.987448 |
| 21282 | kroA100 | 32231 | 30701 | 30957 | 146503.0298 |
| 14379 | lin105 | 24129 | 22508 | 21860 | 107403.6209 |
| 29437 | kroB200 | 67405 | 57890 | 62478 | 303044.6882 |

Table 5. P-values of Wilcoxon signed-rank test for each pair of the methods reported in Table 4.

| | GA | CSGA | MPGA | PSO |
|---|---|---|---|---|
| GA | - | 0.0137 | 0.0756 | 0.0020 |
| CSGA | - | - | 0.1309 | 0.0020 |
| MPGA | - | - | - | 0.0020 |
| PSO | - | - | - | - |

The parameters of the GA are the same as those used in the first set of experiments, except for the cross-breeding rate and cross-breeding interval, where the following values are set: CI=50, CR=10. As these factors have an impact on the proposed algorithm. However, the number of populations of the MPGA was 2. We performed experiments on ten TSP instances, each of which has a known optimal solution. These instances include att48, eil51, berlin52, KroA100, lin105, bier127, ch130, ch150, kroB200 and a280. Table 4 displays the results of the CSGA in comparison to the traditional methods: GA and MPGA. We increased the number of individuals for cross-breeding and at the same time reduced the cross-breeding interval. The aim is to study the effect of these variables on the diversity of the solutions. Consequently, these variables influence the quality of the results.

As seen in Table 4, by contrasting the results found in this table, the proposed CSGA has clearly been able to give more satisfactory outcomes than GA, which is evident from the improvement in the quality of the solutions obtained for 8 instances.

Table 4 demonstrates that the proposed CSGA outperforms the MPGA. Specifically, the CSGA algorithm yielded better results than the MPGA algorithm in 7 instances: a280, att48, bier127, ch130, ch150, kroA100 and kroB200.

The Wilcoxon signed-rank test is a non-parametric statistical test that is used to compare two related samples and determine whether there are statistically significant differences between them. Table 5 shows the p-values from the Wilcoxon signed-rank test for every method pair. In this context, we're comparing the performance of several optimization algorithms on TSP instances.

A p-value is a measure of evidence against a null hypothesis, which in our case is that there are no statistically significant differences in the performance of each pair of methods compared. Researchers typically employ a significance level (alpha) to evaluate whether a p-value is statistically significant or not. The most common alpha levels are 0.05 and 0.01. Here, we considered statistically differences significant if the p-value is less than or equal to alpha=0.05.

Table 4 shows that the proposed CSGA surpasses PSO in all TSP instances tested. The p-value of 0.0020, which is less than 0.05, supports this conclusion, suggesting statistically significant differences between CSGA and PSO. In most cases, the proposed CSGA outperforms classic GA and somewhat outperforms MPGA. The p-value between CSGA and GA is 0.0137, which is smaller than (alpha = 0.05), indicating that the differences are statistically significant. The p-value between CSGA and MPGA, on the other hand, is 0.1309, which is not statistically significant at alpha = 0.05. This is owing to the difference being insignificant, despite being in favour of the proposed CSGA.

It is worth noting that when addressing the TSP problem, the GA produced much better results than the PSO. This conclusion is confirmed further by references [29, 46], which compare the performances of PSO and GA. According to their findings, PSO has quicker computing performance, although GA produces shorter optimized pathways. Also, GA is a better option for dealing with TSP, particularly when time is not a big concern according to [29, 46]. As a result, when handling the TSP issue, the proposed approach outperforms typical GAs and MPGA, as well as one of the well-known optimization methods (PSO). This highlights the ability of the proposed approach to improve the efficiency of GAs in solving the TSP problem, which is a typical example of an optimization problem. Furthermore, this enhancement may be relevant to a broader range of optimization problems; however, further work is required, which is beyond the scope of this paper. The average convergence of GA, CSGA, PSO and MPGA on 10 TSP instances is also shown in Figure 9.

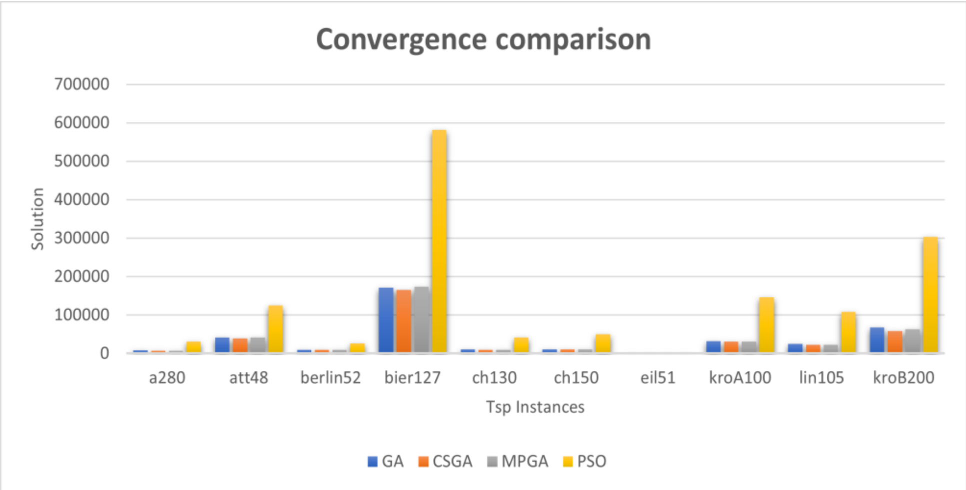


Figure 9. Average convergence of GA and CSGA for big instances from TSP (class2) [9].

It is commonly known that the parameters of GAs have a substantial impact on the results of earlier studies. There isn’t a single, best option for every parameter that may be used in every TSP instance. As a result, adjusting these settings becomes problem-specific [22]. However, the initialization of the starting population is one of the most important parameters. This procedure guarantees that the GA has a good starting point instead of starting from scratch, which entails initializing random solutions. When solving a TSP issue, applying an approximation technique or a heuristic solution during the initialization stage improves the GAŠs performance and speeds up its convergence to better solutions [5].

Consequently, we carried out several experiments utilizing Iterative Approximate Methods for Solving TSP (IAMTSP+), a recent initialization technique primarily intended to offer a heuristic solution for TSP, as detailed in [8]. We also initialized the MPGA and the GA to ensure a fair comparison. This excellent result from IAMTSP+ functions as the Surface population for the proposed CSGA method. We choose to use a different, less advanced initialization method that is based on linear regression (LG) [22] simultaneously. The Cave population responds effectively to this comparatively less sophisticated initialization technique, because the solutions that it provides seem to devolve and are of lower quality than those offered by IAMTSP+. Examples of both techniques applied to cities created at random are shown in Figure 10.

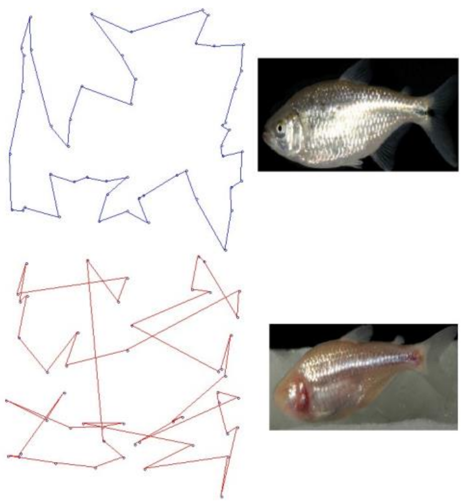

Figure 10. Visualization of the performance of the initialization methods. Top/Surface: IAMTSP+) and Bottom/Cave: LG. Both applied on the same randomly generated TSP.

As anticipated, all methods performed better when initialized with the IAMTSP+ method, as Table 7 illustrates. Of them, CSGA is the most effective method, outperforming the other methods in five TSP instances, yielding outcomes that are on par with the other approaches and obtaining the least average approximation to the optimal solutions. It is worth noting that we used the same parameters in Table 2, except for the CR and CI, which were set to 1. This adjustment helped prevent the Surface population from completely dominating the Cave population, thereby avoiding premature convergence.

This exceptional result can be described to CSGA's novel methodology, which makes use of two separate populations: fully evolved surface fish and less evolved cave fish allowing for more diversity. Diversity is introduced by cross-breeding such populations and then separating their offspring, sending one to the surface and the other to the cave. Through this process, some less-developed genes taken to the surface by the offspring have the opportunity to breed with those in the cave. With time, superior genes from the surface benefit the population residing in caves, while useful genes from the surface can accelerate solution development in the cave, but may reach a local optimum solution without the diversity provided by the offspring that newly inhabited the Surface population. This approach fosters a broader exploration of the search space, contributing to the algorithm's effectiveness. Figure 11 illustrates the performance of the proposed CSGA with IAMTSP+ and LG compared to the same TSP instance (kroA100) shown in Figure 8.

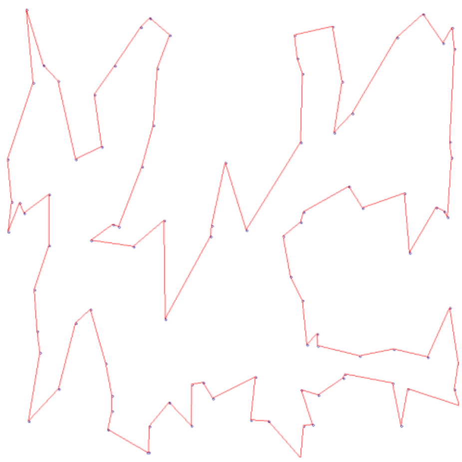

Figure 11. Visualization of the performance of the CSGA when using both initialization methods, IAMTSP+ and LG, applied on kroA100.

Although the proposed CSGA performed reasonably well, in five instances it just marginally outperformed other approaches and in five other cases it was not even close to the best. As seen from the p-values of the Wilcoxon test in Table 7, these show no significant differences. As shown by [8], this phenomenon can be linked to the effectiveness of the initialization process, which autonomously produces near-optimal solutions without requiring a meta-heuristic.

Table 6. Performance comparison of the CSGA, GA and MPGA using initial population (IAMTSP+).

| Instance | MPGA | GA | CSGA | Optimal | App.MPGA | App.GA | App.CSGA |
|---|---|---|---|---|---|---|---|
| a280 | 2984 | 2872 | **2803** | 2579 | 1.157037611 | 1.11361 | 1.086855 |
| att48 | 34414 | 34334 | **33607** | 10628 | 3.238050433 | 3.230523 | 3.162119 |
| berlin52 | 8065 | 7946 | **7748** | 7542 | 1.069345001 | 1.053567 | 1.027314 |
| bier127 | **128142** | 129215 | 128743 | 118282 | 1.083360106 | 1.092432 | 1.088441 |
| ch130 | 6521 | **6418** | 6491 | 6110 | 1.067266776 | 1.050409 | 1.062357 |
| ch150 | 7208 | 7007 | **6954** | 6528 | 1.104166667 | 1.073376 | 1.065257 |
| eil51 | 438 | **435** | 437 | 426 | 1.028169014 | 1.021127 | 1.025822 |
| kroa100 | **22057** | **22057** | 22274 | 21282 | 1.03641575 | 1.036416 | 1.046612 |
| lin105 | 18062 | 15523 | **15485** | 14379 | 1.256137423 | 1.07956 | 1.076918 |
| krob200 | 32149 | **32025** | 33390 | 29437 | 1.092128953 | 1.087917 | 1.134287 |
| Sum | 260040 | 257832 | 257932 | 217193 | 13.13207773 | 12.83894 | 12.77598 |
| Avg. | 26004 | 25783.2 | 25793.2 | 21719.3 | 1.313207773 | 1.283894 | 1.277598 |

Table 7. P-values of Wilcoxon signed-rank test for each pair of the methods reported in Table 6. Differences are shown in the upper diagonal and p-values are shown in the lower diagonal.

| | GA | CSGA | MPGA |
|---|---|---|---|
| GA | | -10 | -220.8 |
| CSGA | 0.6953125 | | -210.8 |
| MPGA | 0.09720109 | 0.4921875 | |

Although the CSGA shows promise for use, not only in the TSP, but also in several related fields, including urban planning, networking, transportation planning, and location-based services, it has the following limitations:

- Solution quality: Using more diverse selection techniques, such as tournament selection, might improve the quality of the solutions even further. More gains could come by tailoring cross-over and mutation operators to the unique features of the proposed CSGA. Addressing these areas of enhancement could pave the way for future research aimed at boosting the performance of the proposed method.
- Separated offspring: In this version of the proposed method, the appearance of offspring on the surface might significantly increase the number of the Surface population, which could lead to memory issues. By keeping the Surface population at a consistent size, this restriction can be lessened. Appropriate actions must be taken as these offspring increasingly converge towards the initial IAMTSP+ solutions. Examples of workable solutions would be coming up with random new solutions or initiating the RG process over again.
- Manual parameter selection: The CSGA's settings were determined by hand without optimization, allowing for future enhancements *via* the application of adaptive parameter-management approaches. The exploration of adaptive parameter-tuning procedures points to a promising future-research direction.

All of these difficulties highlight the need for additional studies to overcome them and improve the efficacy of the proposed CSGA approach.

## 5. CONCLUSIONS

In this study, we introduced a novel multi-population method for the GAs called the Cave Surface Genetic Algorithm (CSGA). Inspired by the evolutionary mechanism observed in cavefish, this algorithm incorporates a secondary population to maintain diversity in the primary population. The cross-breeding mechanism ensures the preservation of a diversified population. The CSGA was applied to various instances of the Traveling Salesman Problem (TSP).

The experimental results show that the proposed CSGA outperforms classical GAs, MPGAs and PSOs in terms of solution quality across the majority of benchmark TSP instances. Nonetheless, limitations must be acknowledged. The parameter choices for the CSGA were selected by hand without optimization, giving the possibility for prospective improvements through the use of adaptive parameter-management techniques. The investigation of adaptive parameter-tuning strategies indicates a promising future research direction.

Extending the scope of our testing to include varied issue domains will not only provide significant insights, but will also further validate the efficacy of the CSGA approach. Furthermore, digging into multi-objective optimization issues has the potential to greatly expand the applicability of our approach. These enhancements and extensions will be the key points of our future-research efforts.

## ACKNOWLEDGEMENTS

The authors genuinely appreciate the reviewers' voluntary efforts and are grateful for their valuable insights.

## REFERENCES

[1] W. F. Abd-El-Wahed, A. A. Mousa and M. A. El-Shorbagy, "Integrating Particle Swarm Optimization with Genetic Algorithms for Solving Nonlinear Optimization Problems," Journal of Computational and Applied Mathematics, vol. 235, no. 5, pp. 1446-1453, 2011.

[2] N. Al-Milli, A. Hudaib and N. Obeid, "Population Diversity Control of Genetic Algorithm Using a Novel Injection Method for Bankruptcy Prediction Problem," Mathematics, vol. 9, no. 8, p. 823, 2021.

[3] M. A. Albadr, S. Tiun, M. Ayob and F. Al-Dhief, "Genetic Algorithm Based on Natural Selection Theory for Optimization Problems," Symmetry, vol. 12, no. 11, p. 1758, 2020.

[4] E. Alkafaween and A. B. A. Hassanat, "Improving TSP Solutions Using GA with a New Hybrid Mutation Based on Knowledge and Randomness," Communications-Scientific Letters of the University of Zilina, vol. 22, no. 3, pp. 128-139, 2020.

[5] E. Alkafaween, A. B. A. Hassanat and S. Tarawneh, "Improving Initial Population for Genetic Algorithm Using the Multi Linear Regression Based Technique (MLRBT)," Communications-Scientific Letters of the University of Zilina, vol. 23, no. 1, pp. E1-E10, 2021.

[6] E. O. Alkafaween, "Novel Methods for Enhancing the Performance of Genetic Algorithms," arXiv preprint, arXiv: 1801.02827, 2018.

[7] E. Alkafaween, S. Elmougy, E. Essa and A. Hassanat, "An Efficiency Boost for Genetic Algorithms: Initializing the GA with the Iterative Approximate Method for Optimizing the Traveling Salesman Problem - Experimental Insights," Applied Sciences, vol. 14, no. 8, p. 3151, 2024.

[8] E. Alkafaween, S. Elmougy, E. Essa, S. Mnasri, A. S. Tarawneh and A. Hassanat, "IAM-TSP: Iterative Approximate Methods for Solving the Traveling Salesman Problem," Int. J. of Advanced Computer Science and Applications (IJACSA), vol. 14, no. 11, DOI: 10.14569/IJACSA.2023.0141143, 2023.

[9] M. Bradic, P. Beerli, F. J. García-de León, S. Esquivel-Bobadilla and R. L. Borowsky, "Gene Flow and Population Structure in the Mexican Blind Cavefish Complex (Astyanax Mexicanus)," BMC Evolutionary Biology, vol. 12, Article no. 9, pp. 1-17, 2012.

[10] C.-M. Chen, S. Lv, J. Ning and J. Ming-Tai Wu, "A Genetic Algorithm for the Waitable Time-varying Multi-depot Green Vehicle Routing Problem," Symmetry, vol. 15, no. 1, p. 124, 2023.

[11] E. C. Osuna, Theoretical and Empirical Evaluation of Diversity-preserving Mechanisms in Evolutionary Algorithms: On the Rigorous Runtime Analysis of Diversity-preserving Mechanisms in Evolutionary Algorithms, PhD Thesis, University of Sheffield, 2018.

[12] K. A. De Jong, "An Analysis of the Behavior of a Class of Genetic Adaptive Systems," Technical Report, University of Michigan, 1975.

[13] M. Dong and Y. Wu, "Dynamic Crossover and Mutation Genetic Algorithm Based on Expansion Sampling," Proc. of the Int. Conf. on Artificial Intelligence and Computational Intelligence (AICI 2009), Proceedings 1, pp. 141-149, Shanghai, China, Springer, November 7-8, 2009.

[14] H. Du, Z. Wang, W. Zhan and J. Guo, "Elitism and Distance Strategy for Selection of Evolutionary Algorithms," IEEE Access, vol. 6, pp. 44531-44541, 2018.

[15] Y. Elipot, H. Hinaux, J. Callebert and S. Rétaux, "Evolutionary Shift from Fighting to Foraging in Blind Cavefish through Changes in the Serotonin Network," Current Biology, vol. 23, no. 1, pp. 1-10, 2013.

[16] Y. Fu, G. Tian, Z. Li and Z. Wang, "Parallel Machine Scheduling with Dynamic Resource Allocation *via* a Master-Slave Genetic Algorithm," IEEJ Transactions on Electrical and Electronic Engineering, vol. 13, no. 5, pp. 748-756, 2018.

[17] J. B. Gross, "The Complex Origin of Astyanax Cavefish," BMC Evolutionary Biology, vol. 12, no. 1, pp. 1-12, 2012.
[18] D. Gupta and S Ghafar, "An Overview of Methods Maintaining Diversity in Genetic Algorithms," Int. J. of Emerging Technology and Advanced Engineering, vol. 2, no. 5, pp. 56-60, 2012.
[19] S. Han and L. Xiao, "An Improved Adaptive Genetic Algorithm," Proc. of the 2022 Int. Conf. on Information Technology in Education and Management Engineering (ITEME2022), SHS Web of Conf., vol. 140, p. 01044, EDP Sciences, 2022.
[20] E.-ul Haq, I. Ahmad, A. Hussain and I. M. Almanjahie, "A Novel Selection Approach for Genetic Algorithms for Global Optimization of Multimodal Continuous Functions," Computational Intelligence and Neuroscience, vol. 2019, Article ID 8640218, pp. 1-14, 2019.
[21] A. Hassanat et al., "Choosing Mutation and Crossover Ratios for Genetic Algorithms: A Review with a New Dynamic Approach," Information, vol. 10, no. 12, p. 390, 2019.
[22] A. B. Hassanat et al., "An Improved Genetic Algorithm with a New Initialization Mechanism Based on Regression Techniques," Information, vol. 9, no. 7, p.167, 2018.
[23] A. B. A. Hassanat, "Enhancing Genetic Algorithms Using Multi Mutations: Experimental Results on the Travelling Salesman Problem," Int. Journal of Computer Science and Information Security, vol. 14, no. 7, p. 785, 2016.
[24] A. B. A Hassanat and E. Alkafaween, "On Enhancing Genetic Algorithms Using New Crossovers," Int. Journal of Computer Applications in Technology, vol. 55, no. 3, pp. 202-212, 2017.
[25] J. H. Holland, Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence, no. 53, ISBN: 9780262581110, MIT Press, 1992.
[26] A. Hussain and Y. S. Muhammad, "Trade-off between Exploration and Exploitation with Genetic Algorithm Using a Novel Selection Operator," Complex & Intelligent Systems, vol. 6, no. 1, pp. 1-14, 2020.
[27] S. Katoch, S. S. Chauhan and V. Kumar, "A Review on Genetic Algorithm: Past, Present and Future," Multimedia Tools and Applications, vol. 80, pp. 8091-8126, 2021.
[28] B. Koohestani, "A Crossover Operator for Improving the Efficiency of Permutation-based Genetic Algorithms," Expert Systems with Applications, vol. 151, p. 113381, DOI: 10.1016/j.eswa.2020.113381, 2020.
[29] L. Kou, J. Wan, H. Liu, W. Ke, H. Li, J. Chen, Z. Yu and Q. Yuan, "Optimized Design of Patrol Path for Offshore Wind Farms Based on Genetic Algorithm and Particle Swarm Optimization with Traveling Salesman Problem," Concurrency and Computation: Practice and Experience, vol. 36, no. 2, p. e7907, DOI: 10.1002/cpe.7907, 2023.
[30] S. Mahmoudinazlou and C. Kwon, "A Hybrid Genetic Algorithm for the Min-max Multiple Traveling Salesman Problem," Computers & Operations Research, vol. 162, p. 106455, DOI: 10.1016/j.cor.2023.106455, 2024.
[31] S. Malik and S. Wadhwa, "Preventing Premature Convergence in Genetic Algorithm Using DGCA and Elitist Technique," Int. Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 6, pp. 410-418, 2014.
[32] E. Simona Nicoară, "Mechanisms to Avoid the Premature Convergence of Genetic Algorithms," Petroleum-Gas University of Ploiesti Bulletin, Mathematics-Informatics-Physics Series, vol. 61, no. 1, 2009.
[33] R. Ohira, M. S. Islam, J. Jo and B. Stantic, "LCS Based Diversity Maintenance in Adaptive Genetic Algorithms," Proc. of the 16th Australasian Conf. on Data Mining (AusDM 2018), pp. 56-68, Bahrurst, NSW, Australia, November 28-30, 2018, Springer, 2019.
[34] E. C. Osuna and D. Sudholt, "On the Runtime Analysis of the Clearing Diversity-preserving Mechanism," Evolutionary Computation, vol. 27, no. 3, pp. 403-433, 2019.
[35] H. M. Pandey, A. Chaudhary and D. Mehrotra, "A Comparative Review of Approaches to Prevent Premature Convergence in GA," Applied Soft Computing, vol. 24, pp. 1047-1077, 2014.
[36] T. Park and K. Ryel Ryu, "A Dual Population Genetic Algorithm with Evolving Diversity," Proc. of the 2007 IEEE Congress on Evolutionary Computation, pp. 3516-3522, Singapore, 2007.
[37] P. V. Paul et al., "A New Population Seeding Technique for Permutation-coded Genetic Algorithm: Service Transfer Approach," Journal of Computational Science, vol. 5, no. 2, pp. 277-297, 2014.
[38] B. R. Rajakumar and A. George, "APOGA: An Adaptive Population Pool Size Based Genetic Algorithm," AASRI Procedia, vol. 4, pp. 288-296, DOI: 10.1016/j.aasri.2013.10.043, 2013.
[39] G. Reinelt, "TSBLIB, 1996" ftp to softlib.rice.edu, 2023.
[40] X. Shi, W. Long, Y. Li, D. Deng and Y Wei, "Research on the Performance of Multi-population Genetic Algorithms with Different Complex Network Structures," Soft Computing, vol. 24, pp. 13441-13459, 2020.
[41] E. Shojaedini, M. Majd and R. Safabakhsh, "Novel Adaptive Genetic Algorithm Sample Consensus," Applied Soft Computing, vol. 77, pp. 635-642, 2019.
[42] M. Srinivas and L. M. Patnaik, "Adaptive Probabilities of Crossover and Mutation in Genetic

Algorithms," IEEE Transactions on Systems, Man and Cybernetics, vol. 24, no. 4, pp. 656-667, 1994.
[43] B. A. Stahl et al., "Manipulation of Gene Function in Mexican Cavefish," Journal of Visualized Experiments (JoVE), vol. 146, p. e59093, 2019.
[44] P. A. Vikhar, "Evolutionary Algorithms: A Critical Review and Its Future Prospects," Proc. of the 2016 Int. Conf. on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), pp. 261-265, Jalgaon, India, 2016.
[45] D. Whitley, S. Rana and R. B. Heckendorn, "The Island Model Genetic Algorithm: On Separability, Population Size and Convergence," J. of Computing and Inf. Technolo., vol. 7, no. 1, pp. 33-47, 1999.
[46] Z. Wu, "A Comparative Study of Solving Traveling Salesman Problem with Genetic Algorithm, Ant Colony Algorithm and Particle Swarm Optimization," Proc. of the 2020 2nd Int. Conf. on Robotics Systems and Vehicle Technology, pp. 95-99, Xiamen, China, 2020.
[47] W. Xu et al., "Optimization Approaches for Solving Production Scheduling Problem: A Brief Overview and a Case Study for Hybrid Flow Shop Using Genetic Algorithms," Advances in Production Engineering & Management, vol. 17, no. 1, pp. 45-56, 2022.
[48] Y. Xue, H. Zhu, J. Liang and A. Sşowik, "Adaptive Crossover Operator Based Multi-objective Binary Genetic Algorithm for Feature Selection in Classification," Knowledge-based Systems, vol. 227, p. 107218, DOI: 10.1016/j.knosys.2021.107218, 2021.
[49] S. Yang, "PDGA: The Primal-dual Genetic Algorithm," Design and application of Hybrid Intelligent Systems, pp. 214-223, Chapter in Book: Design and Application of Hybrid Intelligent Systems, IOS Press, 2003.
[50] P. Zhang et al., "A Genetic Algorithm with Jumping Gene and Heuristic Operators for Traveling Salesman Problem," Applied Soft Computing, vol. 127, p. 109339, DOI: 10.1016/j.asoc.2022.109339, 2022.
[51] G. Zhou et al., "Location Optimization of Electric Vehicle Charging Stations: Based on Cost Model and Genetic Algorithm," Energy, vol. 247, p. 123437, DOI: 10.1016/ j.energy.2022.123437, 2022.

**ملخص البحث:**

الخوارزميات الجينية هي خوارزميات بحثٍ مبنية على الجوانب الجينية للمجتمعات ومفهوم الانتخاب الطّبيعي. ويعدّ الحفاظ على تنوّع المجتمع أمراً حاسماً في الخوارزميات الجينية لضمان الفحْص الشّامل والتّخفيف من خطر الالتقاء قبل الأوان. وتجدر الإشارة إلى أنّ الالتقاء السّريع في اتجاه القيم المثالية المحلية يشكّل أحد أبرز التّحدّيات التي تواجه تطبيق الخوارزميات الجينية.

ولعلاج هذه المسألة، نقدّم في هذه الورقة خوارزميةً جينيةً تسمّى خوارزمية "الكَهْف/السَّطْح"، وتمثّل طريقةً بديلةً قائمةً على الخوارزمية الجينية مزدوجة المجتمع ومستوحاة من التّنوّع الجيني الملاحَظ في سمكة الكهف المكسيكية. ومن خلال الإكثار المتبادل بين المجتمعات، تعمل الطّريقة المقترحة على زيادة التّنوّع عبر مجتمعٍ ثانوي (مجتمع الكهف) وتسهيل تبادل المعلومات بين المجتمعات؛ مقاوِمةً بذلك مشكّلة الالتقاء قبل الأوان.

وقد تمّ إجراء العديد من التّجارب مع الاستفادة من حالاتٍ مرجعيةٍ لما يُعرف بـ "مشكلة البائع المتجول" (TSP) جرى الحصول عليها من مكتبةٍ معروفةٍ لحالاتٍ تتعلّق بتلك المشكلة. وقد بينت نتائج التّجارب أنّ الخوارزمية الجينية المقترحة في هذه الدّراسة والمسماة خوارزمية "الكَهْف/السَّطْح" تفوّقت على الخوارزميات الجينية الكلاسيكية وغيرها من الخوارزميات الجينية الّتي تستخدم تقنيات الحِفاظ على التّنوّع، من حيث إعطاء حلول واعدة للتّحدّيات المتعلّقة بتطبيقات الخوارزميات الجينية.

This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).