Technium Vol. 8, pp.1-11 (2023)
ISSN: 2668-778X
www.techniumscience.com

Building a Tool for Optimal Test Cases Selection using Artificial Intelligence Techniques

Shahbaa I. Khaleel[1], **Raghda Anan Alghadanfary**[2]
[1,2]Department of Software, College of Computer Science and Mathematics, Mosul University, Iraq

shahbaaibrkh@uomosul.edu.iq

ABSTRACT: Software testing is an important process for detecting errors in programs and reducing the risks of their use. With the rapid expansion of the software industry and the heavy dependence on increasingly popular and frequently used programs, there is a necessary need to use software testing techniques that are efficient, scalable, applicable, and effective in detecting errors. In this research, a tool was built that selects the optimal test cases using artificial intelligence techniques. The crow search algorithm was used to select test cases, and after modifications and improvements were made to the algorithm, the improved crow search algorithm was proposed, which generates and selects test cases that achieve the basic paths of the program, depending on the hybridization between the criterion of close to boundary value and branch coverage in calculating the fitness function, and relying on the crow's awareness probability value. In addition, the genetic algorithm was used for test case prioritization.

Keywords. Software testing, Test case selection and prioritization, Artificial intelligence, Crow search algorithm.

1. INTRODUCTION

Software testing is a means of discovering errors expected to occur or occurring in the system and helps to identify and correct errors, defects, malfunctions and failures in the system. Many techniques and strategies have emerged since the concept of software development emerged and the goal of testing is to make software quality as efficient as possible. Among the most important techniques used in testing are: black box testing, white box testing, and gray box testing. Among the most important strategies or phases for software testing are: unit testing, integration testing, system testing, and acceptance testing [1].

Software testing is an essential process in software development because it verifies whether the system meets user requirements and specification. Application testing can be done manually or automatically using software testing tools. The tester acts as the end user and tests the correct behavior of most of the application's features so manual testing is time consuming and demanding and does not always get rid of all bugs effectively but it is an excellent choice for small businesses that do not have sufficient financial resources for automated systems. Manual testing and automated testing are two ways to perform software testing [2].

The automated test improves the accuracy of the results and increases the speed of error detection, and thus is considered more effective when it comes to time, cost, and usability. At the same

time, the automated test has many disadvantages. It requires great effort and time to choose the appropriate tool, as well as knowledge of how this tool works, also the process of purchasing the tool is expensive [3].

Artificial intelligence is the study of how to make computers perform intelligent tasks that in the past could only be performed by humans. Artificial intelligence has developed rapidly recently and changed people's lifestyles. The development of artificial intelligence has become an important method for countries all over the world. Many countries have promoted the dissemination of techniques. Main in order to take the country to a new reality of development and modernity. Artificial intelligence has become an important field of research in science and technology as major companies such as Google, Microsoft and IBM are committed to artificial intelligence and apply artificial intelligence in many fields [4].

Artificial intelligence is one of the branches of computer science and includes the development of computer programs that perform tasks that require human intelligence to implement them. AI algorithms can address learning, cognition, problem solving, language understanding, or logical reasoning. Artificial intelligence is used in many ways in the modern world, from personal assistants to self-driving cars. Artificial intelligence is developing very quickly, and science fiction depicts artificial intelligence in the form of robots that are as close as possible to humans. Artificial intelligence is also called machine intelligence, because it is an intelligence shown by machines, in contrast to the natural intelligence shown by humans and other organisms [5][6]. Among the most important areas in which artificial intelligence is used are philosophy, computer science, psychology, neuron science, biology, mathematics and sociology [7].

This paper is organized as follows: Section 1 gives an introduction of software testing process and artificial intelligence techniques, Section 2 is helpful to understand the background of related works, Section 3 explains the proposed system and the algorithm that used in this work, Section 4 shows the results of the proposed technique and the last section concludes the paper and followed by references.

2. RELATED WORKS

There are many studies and works in the field of reducing the number of test cases and assigning priority to them. In this section, some of these works are summarized as follows:

In 2014, Neha, Shaveta, and Paramjeet used the improved ant colony algorithm in order to reduce the number of test cases as well as assign priority for them. Then they improved this algorithm to get better results, as they used the Average Percentage of Fault Detected APFD metric in order to find the best among the algorithm after improvement and the algorithm before improvement. They concluded in their work that the improved algorithm had better results, as the value of the APFD was equal to 0.91, while in the algorithm before the improvement, its value was 0.67 [8].

In the year 2017, Jagpuneet and Rumanjeet used a hybrid technique in their work. This technique combines two techniques, the first is adaptive, and the second is the genetic algorithm. The adaptive technique priritizes test cases by arranging them from cases with higher priority to cases with lower priority, and they stored the results of this process, as the test cases were arranged using a set of operations, which are parental selection, crossover, and mutation [9].

In 2017, Tamim, Erum, Khurum, and Shaftab used the swarm algorithm for prioritizing test cases, where they implemented the algorithm using three programs and compared the performance efficiency of the algorithm with another random technique. The results showed that the proposed algorithm was more efficient in detecting errors in the early stages of the software development life cycle [10].

In 2018, Deepak, Sheikh, and Preetam used a hybrid intelligent technique that combines the ant colony algorithm with the genetic algorithm. They first formed the test suites and arranged them according to their priority, depending on their degree of change, importance, complexity, execution time, error rate, and the amount of coverage [11].

In the year 2019, Ms. Manaswini and Rama used the cat swarm algorithm to prioritize the test cases and their order based on the fitness function and used metrics such as the amount of coverage, the percentage of error detection and the execution time of the test cases. In their work, they demonstrated the efficient performance of the algorithm, as it detected 4 errors within 6 seconds for one of the test cases [12].

In the year 2020, Tina compared the performance of the swarm algorithm with the performance of the genetic algorithm GA in prioritizing test cases, depending on the execution time. In her work, she concluded that GA and Ant Colony Optimization algorithm ACO are better in their results than other algorithms. She also concluded that the execution time of GA is very close to the execution time of ACO. They are much better than other algorithms that may consume five times the time it takes to implement them. The genetic algorithm took 1.0 ms while the ACO took 1.1 ms when both were implemented with the same test set [13].

3. THE PROPOSED SYSTEM

In this research, a system was proposed that analyses paths of the program and obtains the optimal test cases for the source code for which the testing process is intended, the Crow Search Algorithm was used to generate and select the test cases that achieve the basic paths of the program by using two fitness functions, which are the fitness function based on the close to boundary value criterion and the fitness function based on the branch coverage criterion. The performance of the algorithm was improved to serve the work for reaching the best results by relying on the hybridization between the two fitness functions and the conclusion of the hybrid fitness function. The work was also improved by relying on the dynamic awareness probability value. Figure 1 shows the general outline of the work steps for the proposed tool.

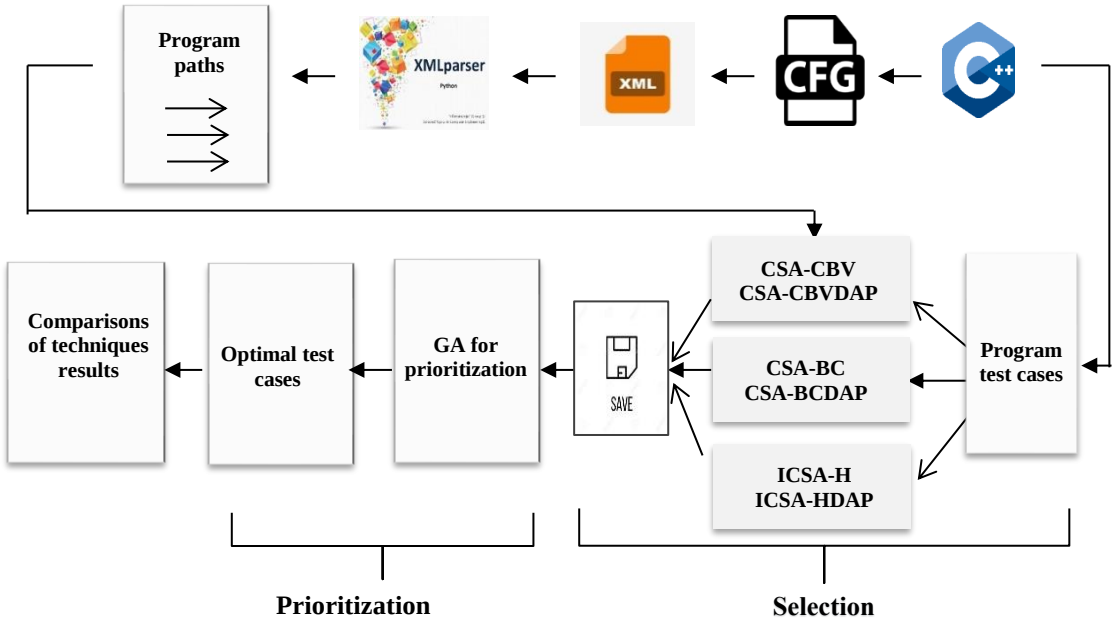

Figure 1 : A general outline of the system

Technium Vol. 8, pp.1-11 (2023)
ISSN: 2668-778X
www.techniumscience.com

4. CROW SEARCH ALGORITHM CSA

It is one of the computational intelligence techniques proposed by Askarzadeh in 2016 that is used to solve complex problems and is inspired by the collective behavior of colonies of crows. A swarm can be considered as a group of individuals or members interacting together using some special behaviours that differ according to the type of swarm to achieve a specific goal. The swarm is represented by groups of individuals, these individuals use simple protected rules to control their movements and interactions with other individuals who are together in the same swarm. The community system in the swarm consists of a group of individuals, and each individual in this swarm is independent in its environment from other individuals, and they all form an integrated system, but they are not subject to specific orders from a leader, and they do not follow a fixed plan, as each bird in the swarm is independent in making its decisions, but it changes its position and moves in accordance with the other birds present with it in the exile of the swarm and it tries to stay close to individuals or other birds and tries not to collide with them [14].

The fitness function is used as a basic element in the optimization algorithms, as it directs the implementation towards the optimal solution. Below is more detail on the fitness functions that were relied upon when implementing the CSA in this work:

❖ Fitness function based on close to boundary value criterion: Since the chances of errors occurring are mostly at boundary values, test cases close to boundary values must be given higher precedence than other test cases. The approach factor to the boundary value *CBV* for the test group *T* is calculated through the following equation [15] :

$$CBV(T) = CBV(t_1) \times CBV(t_2) \times \dots\dots \times CBV(t_n) \quad \text{....(1)}$$

Where *CBV(t)* is the approach factor to the boundary value for the test case *t*.

❖ Fitness function based on the branch coverage criterion: the branch coverage factor is used to select the test case in most of the automated test tools, and it is calculated from the ratio of the number of edges that are covered by the test cases divided by the edges of the Control Flow Graph of the program to be tested [15], its value is calculated according to the following equation:

$$BC(T) = nodes \div edges \quad \text{.....(2)}$$

When applying the algorithm, the following basic steps must be adopted [16]:

1. Parameter values of algorithm are initialized which are adjustable according to the problem to be solved such as swarm size n, reptmax, flight length fl and awareness probability ap.
2. In the pd-dimensional search space, n crows are randomly placed as individuals in the swarm. Each crow finds a solution, and pd is the number of search space variables. The memory mr is initialized for each crow in the swarm because crows do not have experience with the initial values of the first iteration because they will think that their food in the first positions is gone.
3. A position is determined for each crow by finding the objective function, i.e. the fitness function. In this research, the following equation for the fitness function was relied on based on the criterion of close to boundary value:

   $$CBV(T) = CBV(t_1) \times CBV(t_2) \times \dots\dots \times CBV(t_n)$$

   And the following equation was used for the fitness function based on the branch coverage criterion:

   $$BC(T) = nodes \div edges$$

4. The crows in the search space create the new position. Assuming that the crow i wants to create a new position, then this crow chooses a random crow, for example, the crow j, to chase it, monitor and discover how and where this crow hides its food. According to the following equation, the new position of crow i is updated:

$$p^{i,rept+1} = \begin{cases} p^{i,rept} + r_i \times fl^{i,rept} \times \left(m^{j,rept} - p^{i,rept}\right) & \rightarrow \quad r_j \geq ap^{j,rept} \\ a\ random\ position & \rightarrow \quad otherwie \end{cases} \quad \text{.....(3)}$$

Technium Vol. 8, pp.1-11 (2023)
ISSN: 2668-778X
www.techniumscience.com

5. Check the stability of all new crows positions and the crow updates its new position.
6. The value of the fitness function is found for each new position of each crow.
7. The memory of each crow is updated as follows:

$$mr^{i,rept+1} = \begin{cases} p^{j,rept+1} & \rightarrow \quad \left(fl\left(p^{j,rept+1}\right) \geq fl\left(mr^{j,rept}\right)\right) \\ mr^{j,rept} & \rightarrow \quad otherwie \end{cases} \qquad \text{.....(4)}$$

where mr is the crow's memory, p is the crow's position, and fl is the flight length of the crow. The crow updates the new position in its memory if the value of the fitness function for the new position is better than the value of the fitness function for the place that was remembered, i.e. according to the value of the memory position it has.

8. Steps 4 to 7 are repeated until reptmax is reached and the best memory position is reached, which returns to the value of the objective function, which is the solution to the optimization problem, and then the termination condition is reached.

Figure 2 shows the block diagram of the algorithm's work steps in this research, and Figure 3 shows the flowchart of the algorithm.

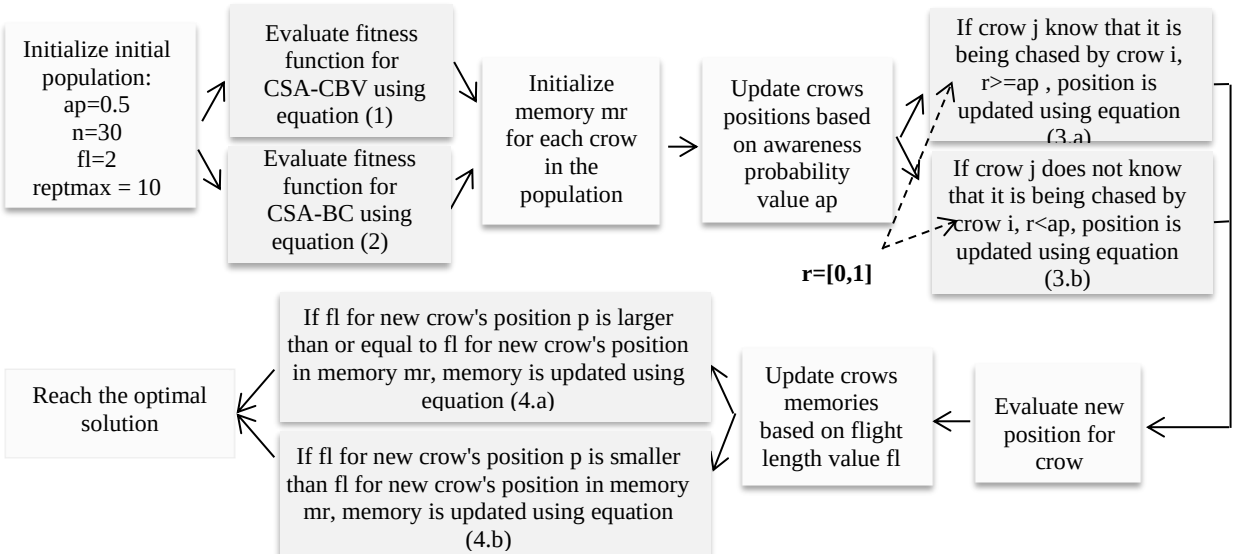

Figure 2 : Block diagram of the algorithm steps

In this research, a set of improvements were made to the features of the standard algorithm to reach the desired goals and to reach the improved algorithm. These features are:

● Fitness function: modifications have been made to the fitness function used to serve the work and improve the performance of the algorithm. The improvement is proposed by performing the hybridization between the two fitness functions that were used in the standard CSA algorithm, based on the following equation:

$$fitness = 1 - \left((Boundary + Bc) \div 2\right) \qquad \text{....(5)}$$

Where *Boundary* is the value of the fitness function dependent on close to boundary value represented by the symbol *CBV* and *Bc* is the value of the fitness function dependent on branch coverage.

● The crow's awareness probability value: It was suggested that an modification be made to this feature to improve the performance of the algorithm in order to obtain better results by making the ap value dynamic, that is, it changes at each generation of of random candidate crows. This value is calculated based on the following equation [17]:

$$DAP_{i,k} = 0.9 \times \left(\frac{F(p_{i,k})}{WV}\right) + 0.1 \quad \ldots.(6)$$

Where, F is the value of the fitness function of the crow, p represents the position of the crow and WV is the worst value found for the fitness function.

The genetic algorithm was used for assigning priority to the test cases obtained from the selection process using the proposed crow search algorithm. Prioritization techniques try to reach the lowest cost in the testing process because it reduces the time spent and the budget and it also consumes less effort in the testing process.

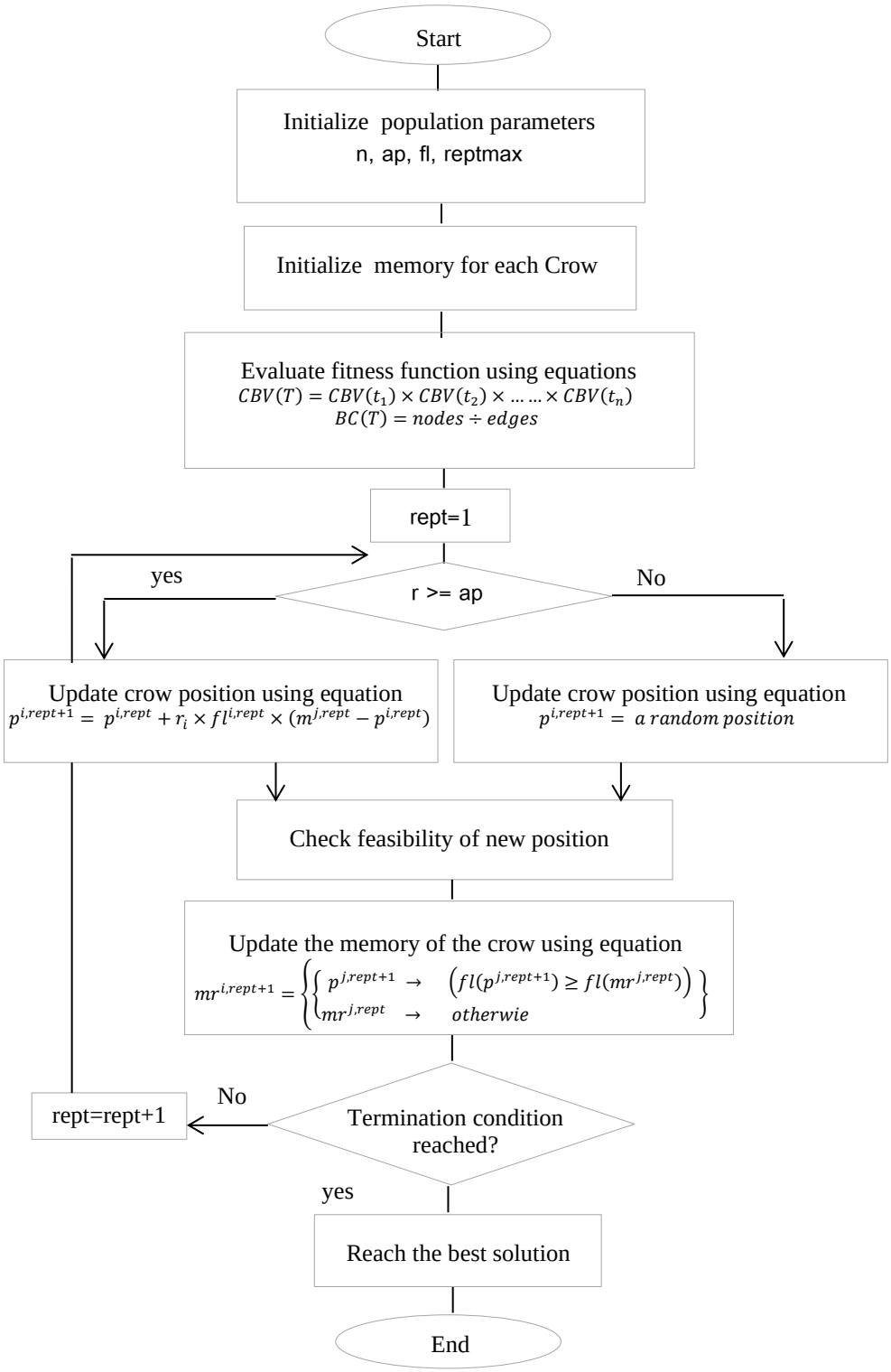

Figure 3 : A flowchart of the algorithm

The genetic algorithm helps to find the optimal arrangement for the test cases, where the genetic information of the chromosomes is used to direct the search to find the best solutions in the search space, and the genetic information, that is, the chromosomes, is represented as a series of test cases. The genetic algorithm includes basic elements, which are the selection of the starting population, the encoding of chromosomes, the mechanism of selection and crossover.

When completing the identification of the basic elements of the genetic algorithm and determining the number of iterations of the algorithm as a condition for termination, its work begins, as the initial generation is formed, which is represented by the positions of the test cases, and they are arranged randomly within the chromosome. Then the fitness function is calculated for each individual in the generation, which is the hybrid fitness function that depends on the two criteria of close to boundary value and the branch coverage, and then two members of the generation are chosen to be parents in the next generation, then the crossover process takes place. The fitness function is calculated for all individuals resulting from the crossover process, and the result is compared with the fitness function for the parents, and on its basis, either the replacement is made between the parents and the new individuals in the community, or it is not, and the process continues until the termination condition that was determined at the beginning is reached, and finally it is obtaining the individuals with the highest fitness function in the community and their ranking is approved and stored as a result of the process of prioritization.

5. RESULTS

Several programs were used to test the proposed system, and they include most of the components of the basic structure of the source code, CaseStudy1 finds the largest number among three numbers using a nested if statement, CaseStudy2 finds the largest number among three numbers using a compound condition, CaseStusy3 collects the main diagonal elements and prints sum using for, CaseStudy4 sums the elements of the main diagonal and prints the sum using nested loops, CaseStudy5 enters student averages and determines whether the student is successful or fail using the while loop, CaseStudy6 enters student averages and calculates the number of successful and fail students, CaseStudy7 determines the student's grade based on the entered student's average using switch and CaseStudy8 calculates the student's average and grade based on entered student's grades.

The algorithm was implemented using the number of crows swarm equal to 30. The number of test cases used before training, the number of iterations, the time it took for the training process, and the test results were shown in the results tables for all cases that were implemented for the following techniques: CSA-CBV, CSA-CBVDAP, CSA- BC, CSA-BCDAP, ICSA-H and ICSA-HDAP.

Table 1 shows the results of the CSA-CBV technique and the improved CSA-CBVDAP when the algorithm iterations are equal, which is 10 iterations, and the time varies, which was measured in seconds. It was observed that when the update equation was used for the value of ap and made it dynamic , it gave better results, as the test cases that achieve the paths for the programs increased within the different case studies.

Table 1 : CSA-CBV & CSA-CBVDAP implementation results

Technique used	Program	Time consumed	No of test cases used for training	No of test suites after selection	No of test suites that achieve all paths
CSA-CBV	CaseStudy1	93.7583	120	13	7
	CaseStudy2	111.6318	90	12	5
	CaseStudy3	58.2213	90	12	6
	CaseStudy4	111.7047	120	11	3

	CaseStudy5	38.4930	90	11	5
	CaseStudy6	53.7831	90	12	6
	CaseStudy7	48.3887	240	10	7
	CaseStudy8	74.4639	240	11	6
CSA-CBVDAP	CaseStudy1	92.2436	120	15	8
	CaseStudy2	90.1678	90	16	7
	CaseStudy3	53.9498	90	15	7
	CaseStudy4	101.4477	120	14	6
	CaseStudy5	36.7083	90	15	8
	CaseStudy6	43.9871	90	17	8
	CaseStudy7	39.5767	240	18	10
	CaseStudy8	57.4990	240	15	9

Table 2 shows the results of the CSA-BC technique and the improved CSA-BCDAP technique for 10 iterations as follows:

Table 2 : CSA-BC & CSA-BCDAP implementation results

Technique used	Program	Time consumed	No of test cases used for training	No of test suites after selection	No of test suites that achieve all paths
CSA-BC	CaseStudy1	91.0562	120	10	2
	CaseStudy2	80.4859	90	8	3
	CaseStudy3	52.3034	90	7	4
	CaseStudy4	85.9777	120	8	3
	CaseStudy5	36.4862	90	7	2
	CaseStudy6	42.1325	90	9	4
	CaseStudy7	47.8532	240	7	3
	CaseStudy8	54.0817	240	9	5
CSA-BCDAP	CaseStudy1	83.9913	120	12	5
	CaseStudy2	76.1817	90	11	6
	CaseStudy3	51.7019	90	12	5
	CaseStudy4	83.1286	120	11	6
	CaseStudy5	35.1245	90	10	4
	CaseStudy6	40.2187	90	12	5
	CaseStudy7	37.4702	240	11	6
	CaseStudy8	49.9913	240	12	6

Despite the badness of this technique, it does not produce many test cases to achieve all the paths, but when updating the ap value and making it dynamic, the work of this technique improved and produced more test cases than the previous one and achieved the paths.

Table 3) shows the results of the ICSA-H, improved ICSA-HDAP technique for 10 iterations as follows:

Table 3 : CSA-H , CSA-HDAP & CSA-HDAPSP implementation results

Technique used	Program	Time consumed	No of test cases used for training	No of test suites after selection	No of test suites that achieve all paths
ICSA-H	CaseStudy1	108.2125	120	20	16
	CaseStudy2	97.1139	90	21	15

	CaseStudy3	67.4981	90	20	15
	CaseStudy4	95.7452	120	20	14
	CaseStudy5	41.0606	90	18	11
	CaseStudy6	43.5620	90	15	12
	CaseStudy7	49.6960	240	20	14
	CaseStudy8	62.5798	240	21	17
ICSA-HDAP	CaseStudy1	99.4880	120	26	24
	CaseStudy2	90.2889	90	25	24
	CaseStudy3	59.9429	90	24	20
	CaseStudy4	90.9233	120	25	22
	CaseStudy5	40.4837	90	26	23
	CaseStudy6	40.6357	90	24	21
	CaseStudy7	37.8030	240	24	20
	CaseStudy8	57.1542	240	25	21
ICSA-HDAPSP	CaseStudy1	89.5355	120	30	30
	CaseStudy2	76.7515	90	30	30
	CaseStudy3	49.7772	90	30	30
	CaseStudy4	82.1856	120	29	29
	CaseStudy5	32.8317	90	29	29
	CaseStudy6	37.3927	90	29	29
	CaseStudy7	37.6132	240	28	28
	CaseStudy8	47.6120	240	27	27

As for this ICSA-H technique, it was noticed that when improving the work by adding a hybrid fitness function, good results were obtained, but its work was also improved by using the dynamic ap value, which is calculated according to Equation No. (6), and it was noted that the number of test cases that achieve all the basic paths of the program to be tested increased.

After the selection process has been performed and suites of test cases have been obtained using the crow search algorithm, the genetic algorithm is used to perform the process of prioritization to those test cases so that the best test cases are obtained in a certain order, when the algorithm is executed on CaseStudy1, 3 test suites were obtained, in CaseStudy2, 3 test suites were obtained, in CaseStudy3, 2 test suites obtained, in CaseStudy4, 4 test suites obtained, in CaseStudy5, 9 test suites obtained, in CaseStudy6, 2 test suites obtained, in CaseStudy7, 2 test suites obtained and in CaseStudy8, 2 test suites obtained.

6. CONCLUSIONS

The standard CSA algorithm was used in the process of test case selection, and it turned out that it does not serve the work adequately. It gave a small number of test cases, and a some of them did not achieve all the basic paths of the program. Therefore, it was proposed to make modifications to the algorithm. When improving the awareness probability value of crow, which is one of the basic parameters The crow search algorithm gave good results in test case selection and took less time than it took when the value of crow awareness probability was constant. For example, in the CaseStudy5 program, during ten iterations, 15 test suites were selected out of 90 test cases, and 8 of them achieved the basic paths of the program, and in 36.70 seconds after it took 38.49 minutes to find 12 test suites, and 6 of them achieve the basic paths of the program. When the improvement was made on the fitness function of the algorithm, which is the main element in the swarm algorithms, it gave good results in selecting test cases. For example, in the CaseStudy3 program, during ten iterations,

20 test cases were selected out of 90 test cases, and 15 of them achieved the basic paths of the program in 67.49 seconds. When relying on the hybrid fitness function and the dynamic value of the crow's awareness probability, the algorithm gave better results. In the CaseStudy1 program, during ten iterations, 26 test suites were selected out of 120 test cases, and 24 of them achieved the basic paths of the program in 99.48 seconds.

7. REFERENCES

1. Jan, S. R., Shah, S. T. U., Johar, Z. U., Shah, Y., & Khan, F. , (2016), "An innovative approach to investigate various software testing techniques and strategies" , *International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Print ISSN*, *23951990*.
2. Gamido, H. V., & Gamido, M. V. , (2019), "Comparative review of the features of automated software testing tools" , *International Journal of Electrical and Computer Engineering*, *9*(5), 4473.
3. Umar, M. A., & Zhanfang, C. , (2019) , "A study of automated software testing: Automation tools and frameworks" , *International Journal of Computer Science Engineering (IJCSE)*, *6*, p. 217-225.
4. Zhang, C., & Lu, Y. , (2021) , "Study on artificial intelligence: The state of the art and future prospects" , *Journal of Industrial Information Integration*, *23*, 100224.
5. Mohammed, Z. , (2019) , "Artificial intelligence definition, ethics and standards" , *Electronics and communications: Law, standards and practice*.
6. Khaleel, S. I., & Anan, R. (2023), "A review paper: optimal test cases for regression testing using artificial intelligent techniques". *International Journal of Electrical and Computer Engineering (IJECE)*, *13*(2), 1803-1816.
7. Ahmed, H. E. , (2018) , "AI Advantages and disadvantages" , *International Journal of Scientific Engineering and Applied Science (IJSEAS)*, 4(4), pp. 22-25.
8. Sethi, N., Rani, S., & Singh, P. , (2014) , "Ants optimization for minimal test case selection and prioritization as to reduce the cost of regression testing" , *International journal of computer applications*, 100(17).
9. Bajwa J. K., Kaur R. , (2017) , " An Adaptive Approach For Test Case Prioritization In Regression Testing Using Improved Genetic Algorithm", *An International Journal Of Engineering Sciences*, Vol: 24, P:8.
10. Ashraf, E., Mahmood, K., Khan, T. A., & Ahmed, S. , (2017) , "Value based PSO test case prioritization algorithm" , *International Journal of Advanced Computer Science and Applications*, *8*(1).
11. Ahmad, S. F., Singh, D. K., & Suman, P. , (2018) , "Prioritization for regression testing using ant colony optimization based on test factors", *Intelligent communication, control and devices*, pp. 1353-1360, Springer, Singapore.
12. Manaswini B., & Reddy A. R. M., (2019), " A Cat Swarm Optimization Based Test Case Prioritization Technique to Perform Regression Testing", *International Journal of Recent Technology and Engineering (IJRTE)*, Vol. 8, No.1, p. 2677-2682.
13. Sachdeva T. , (2020) , " Swarm Intelligence Techniques And Genetic Algorithms For Test Case Prioritization", *International Journal Of Engineering And Advanced Technology (IJEAT)*, Vol: 9, Issue: 4, P:465.

14. Askarzadeh, A. ,(2016), "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm". *Computers & structures*, *169*, 1-12.

15. Harsimran Singh, 2004, " Automatic Generation of Software Test Cases using Genetic Algorithms", *Master thesis In Software Engineering*, *Computer Science & Engineering Department Thapar Institute of Engineering & Technology*, *Patiala*.
16. Gadekallu, T. R., Alazab, M., Kaluri, R., Maddikunta, P. K. R., Bhattacharya, S., & Lakshmanna, K. , (2021) , " Hand gesture classification using a novel CNN-crow search algorithm" , *Complex & Intelligent Systems*, *7*(4), p. 1855-1868.
17. Díaz, P., Pérez-Cisneros, M., Cuevas, E., Avalos, O., Gálvez, J., Hinojosa, S., & Zaldivar, D. , (2018), "An improved crow search algorithm applied to energy problems" , *Energies*, *11*(3), 571.