Technium Vol. 8, pp.60-71 (2023)
ISSN: 2668-778X
www.techniumscience.com

Constructing a Tool for Software Regression Testing Based on Crow Search Method

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

ABSTRACT: The software testing phase is an essential phase of software development. Its aim is to ensure that the program meets the desired requirements of it. Through the testing process, errors are found in the programs so that they can be fixed before deployment, i.e. before being used or delivered to the customer. The program must also be tested after to be published and delivered, this is called regression testing and it is one of the basic activities in software development and it must be done in the software maintenance phase to ensure its reliability. In this research, a tool was built that selects the optimal test cases that are used in the regression testing phase, using artificial intelligence techniques. The Crow Search Algorithm was used in the test case selection, and after modifications and improvements were made to the algorithm, the Improved Crow algorithm was proposed, which generates and selects test cases that achieve the basic paths of the program based on the improved fitness function, the dynamic awareness probability value of the crow, and the spiral search mechanism for the crows. In addition, the genetic algorithm was used for these test cases prioritization.

Keywords: Artificial intelligent techniques, Regression testing, Crow search method

1. INTRODUCTION

Software testing is defined as the process of checking a program in order to detect errors or defects in it. Software is also tested in order to ensure that the program performs its purpose correctly, access to the program, and achieve and maintain its quality, thus verifying that the program is suitable for use. Software testing is an integral phase during the software development life cycle SDLC process. Testing uses about 50% of the time and effort to develop software projects during SDLC. The program is considered incomplete until the testing process is completed. The general purpose of testing is not to prove that the system is free from errors, but rather to give confidence that the system works well before it is installed. If we have an error-free system, it will be believed that the system has not been tested or that it has gone through a bad testing phase [1].

The software testing stage plays an important role in the life cycle of building the system because it leads to a high degree of quality and efficiency, and it is an expensive phase, as it consumes approximately 50% - 80% of the cost allocated to the software development process. Before the system is implemented, it must undergo a testing process in order to ensure that it meets the customer's requirements [2].

Test planning begins during the initial phases of requirements analysis and continues systematically with continuous improvements during software development until the completion of the coding phase and the beginning of executing the test cases. The existence of errors and the cost of conducting these tests, which also require a large amount of time, as well as the cost of finding them

and the cost of correcting them. The test cannot show the absence of defects, but can only detect the presence of defects in software [3].

The term artificial intelligence accurately describes the computational processes, as the process of collecting its ideas has a major impact on the quality and productivity of different types of products and services, as well as the functions that operate within these products and services, their productivity and integration. Some of these ideas relate to programming, artificial life, future outlook, data warehouse, distributed artificial intelligence, expert systems, genetic algorithms, knowledge representation, machine learning, understanding natural languages, neural networks, proof of theories, and computational theories. In other subjects, artificial intelligence is considered an essential tool aimed at developing the world, such as areas related to technology, science, academic fields, health, commerce, management, finance, marketing, economics, marketing management, and law [4].

There are many phases of software testing ST, during which many techniques are used, and among these techniques, the most important of which is the regression testing RT. Regression testing is of great importance because it verifies that the code that was updated recently, whether its previous functions were affected by this modification or not, it does not matter how many times you make changes in the code, but regression testing must be carried out to ensure the stability of the program after the code has been changed [5].

The rest of this research is structured as follows: Section 1 contains an introduction of the importance of the testing phase and artificial intelligence techniques. Section 2 explains regression testing and its techniques. In Section 3 test cases prioritization techniques are reviewed. Section 4 contains the related works. Section 5 and section 6 describes the proposed system and the algorithm that used in this research, In Section 7 the results of the work is presented and finally the conclusion is presented in section 8 then references are listed in section 9.

2. REGRESSION TESTING

Regression testing is the type of testing in which the software is tested when the modification is made in the version under modification to test the new version of the software [6]. Regression testing as one of the main types of testing consumes a lot of time when done manually and it usually checks if the program or application is still working after fixing any error because sometimes after fixing the error, the error percentage in the source code or application increases so to avoid wasted time, a set of automated test suites is created to form a regression testing suite. This also helps in error detection at a very early phase which saves much of the cost of modification as well as effort in later phases [7].

Regression testing techniques run recently completed test cases and then check if the behaviour of the program has changed or re-emerged bugs or problems that were recently fixed and this takes a great deal of effort and time when the program size and complexity increases [8]. The regression testing process involves the use of several techniques and each has its own concept, namely:

- Retest-All technique: In this technique, all test cases in test suites are re-executed, but this type of technique is inefficient because it consumes a lot of time and resources, which makes the regression testing process expensive [9][10].
- Regression Testing Selection technique: This technique is based solely on some metrics that are test case execution cost , size of code also known as code coverage, the ability to find potential errors and modifications in code [11]. In this technique, a part of the test suite is selected for execution instead of selecting the entire test suite for re-execution, where the test cases are categorized into reusable test cases and test cases that are out dated and no longer needed. Reusable test cases are used in scheduled regression cycles while legacy test cases are not used in consecutive cycles [9].
- Test Cases Prioritization technique: In this technique, the test cases are ranked in an order that tries to increase its usefulness in achieving efficiency in performance and increasing the error detection rate, i.e. increasing the speed in noticing errors in the testing process. Increasing the rate

of error detection during testing can provide faster response to the system under test and allow software engineers to start debugging as early as possible [9]. Regression testing is a technique by which changes can be tested using existing test cases. In this technique test cases are given precedence according to their priority which depends on the changes made in the project.

3. TEST CASES PRIORITIZATION TECHNIQUES

Various techniques have been proposed recently for test case prioritization and these techniques are [6][12]:

- Prioritization based on the customer's requirements: The requirements of the customer are taken into account and given a value. Based on these values, the test case for the requirements is checked, and first execute the test cases with the highest value, and then executes the cases with the lowest value.
- Prioritization based on amount of coverage: This technique relies on analysing the coverage area of the source code and measuring the code covered by the test case. Various coverage criteria are considered and the amount of coverage is evaluated and used to determine the precedence of test cases. Coverage-based technology is a white-box testing technique which is a method that tests the internal structures of a program.
- Prioritization based on cost: This technique prioritizes test cases that are based on cost factors such as cost of test cases running, cost of analysis, cost of prioritization, cost of executing and validating test cases. This cost is of two types, the direct cost, which includes test selection, test execution, and results analysis, and the indirect cost, which includes the overhead cost and tool development cost.
- Prioritization based on event logging: This technique prioritizes test cases based on test cases of previous executions in order to increase or decrease the probability that they will be considered in the current test execution.

4. RELATED WORKS

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

Ahlam, Ana, Alisha, and Konain used a technique in 2016 to prioritize test cases for improving and increasing the efficiency of the test suite. They used the Ant Colony Optimization algorithm ACO in their work, where the input set was a set of test cases and it was arranged based on a specific criterion. Through their work, they were able to reduce the time and cost required for the testing process. To measure work efficiency, they used the Average Percentage of Fault Detected APFD metric, whose value for the ordered test cases was equal to 0.633, and its value was 0.533 for the unordered test cases [13].

In their 2017 paper Arvinder and Arun presented an evaluation of the efficiency of two algorithms, Bat algorithm and Cuckoo search algorithm, used in the process of selecting test cases. The consumed time and the percentage of error detection were the two factors used to evaluate the work of the algorithm and in their results it was proved that Cuckoo search algorithm gave better results in terms of efficiency than Bat algorithm while Bat algorithm gave better results in terms of the time taken to implement the algorithm [14].

Technium Vol. 8, pp.60-71 (2023)
ISSN: 2668-778X
www.techniumscience.com

In 2019, Priyanka and Anita used Cuckoo Search Algorithm to prioritize test cases and used the APFD metric in order to compare the performance of Flower Pollination Algorithm FPA with the performance of another traditional algorithm. The results prove that Cuckoo Search Algorithm is better than FDA. they used the Java language with the Eclipse platform to implement the Cuckoo algorithm on two different programs [15]..

In 2019, Muhammad Kh. and Mohd, Dayang, Haza, and Muhammad D. used a technique to improve test cases by prioritizing them using Firefly algorithm. The algorithm after its implementation gave good results, and they verified its efficiency using the APFD metric, and this algorithm also outperformed the Local Beam Search algorithm LBS in terms of execution time [16].

In the year 2020 the Cat Swarm Optimization CSO algorithm was implemented by Arvinder and Richa in order to prioritize test cases. They used to measure the efficiency of the algorithm, the metric Error Detection Rate EDR, and the algorithm included two steps, the first is defining a set of test cases, and the second is an evaluation of the efficiency of these test cases. CSO gave good results compared to other algorithms [17].

5. THE PROPOSED SYSTEM

In this research, a system was proposed that analyses the paths of the program and obtains the optimal test cases for the source code for which the testing process is to be conducted. Conducting the training process where the fitness function based on close to boundary value criterion is combined with the fitness function based on branch coverage criterion, and the work has been improved based on dynamic awareness probability value. Also, the spiral search mechanism has been adopted to update the crows positions in order to improve the performance of the algorithm and to give the best possible results. Figure 1 shows the general outline of the work steps for the proposed tool.

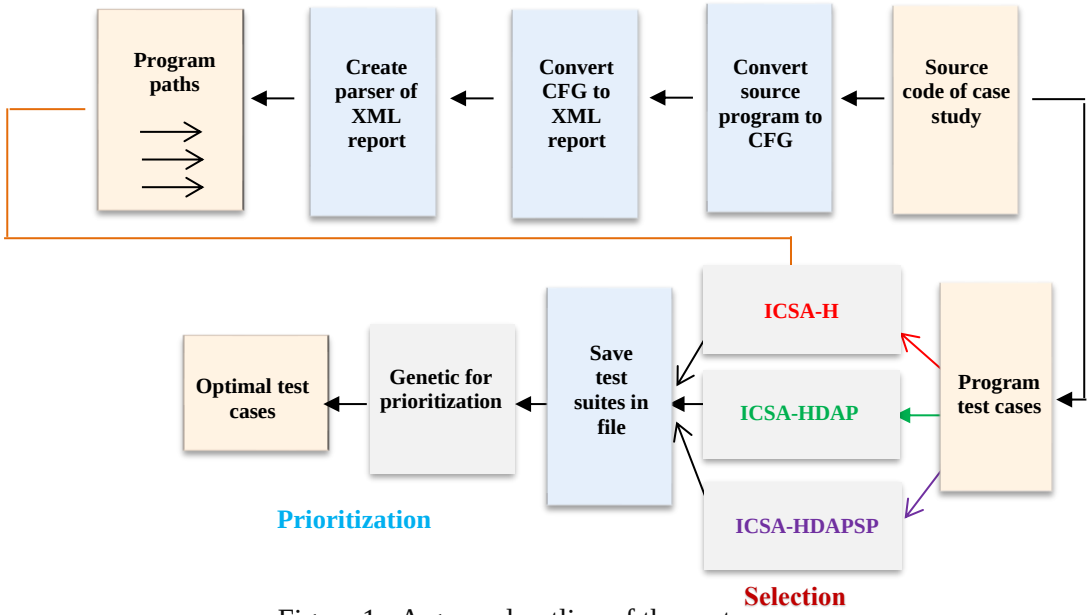

Figure 1 : A general outline of the system

Technium Vol. 8, pp.60-71 (2023)
ISSN: 2668-778X
www.techniumscience.com

6. CROW SEARCH ALGORITHM CSA

The CSA algorithm was proposed by Askar Zadeh. This algorithm came from the collective search mechanism that crows perform to hide their food. Crows are considered among the smartest birds, as they have a larger brain compared to their body size, which is slightly less than that of humans. In addition, they have self-awareness in the mirror test. They are known to be thieves because they steal the food of other birds and use their experience as thieves to anticipate the behaviour of thieves. They are very careful when the crow reports an stealing occurrence. When stealing their food, they move the food to another place, so they can avoid any kind of theft at a later time [18]

The fitness function is used as a basic element in the optimization algorithms, as it directs the implementation towards the optimal solution. In this research, the improved fitness function was used in order to serve the work and improve the performance of the algorithm. The value of this function is calculated based on the following equation:

$$fitness = 1 - \left((B1 + B2) \div 2\right) \quad \text{.....(1)}$$

Where *B1* is the value of the fitness function based on close to boundary value criterion and *B2* is the value of the fitness function based on branch coverage criterion.

When applying the algorithm, there are a set of basic steps that are followed, as follows:

1. Defining and initializing the values of the basic parameters of the algorithm, which are values that change according to the issue to be solved. These values are the size of the swarm n, the maximum number of iterations reptmax, the flight length fl.
2. A number of crows n are placed randomly in the search space pd, which represents the number of variables of the issue to be solved, and the mr crows memory is initialized in the swarm.
3. Crows' positions are evaluated by calculating the fitness function for them, based on the following equation:

$$fitness = 1 - \left((B1 + B2) \div 2\right)$$

4. A random crow is chosen from the search space in order to be chased, then the dynamic awareness probability value DAP of the crow is calculated, which depends on the fitness function that was found. The value of DAP is calculated by the following equation [19]:

$$DAP_{i,k} = 0.9 \times \left(\frac{F(p_{i,k})}{WV}\right) + 0.1 \quad \text{.....(2)}$$

5. Depending on the DAP value from the previous step, the new position of the crows in the search space is updated according to the following equations [20]:

$$p^{i,rept+1} = \begin{cases} D \times exp(b \times l) \times \cos^{(2\pi l)} + gbest & \rightarrow rand \geq DAP \\ \begin{cases} p^{i,rept} + rand & \rightarrow r_j \geq 0.2 \\ p^{i,rept} \times Gaussian(\mu, \sigma) & \rightarrow otherwise \end{cases} & \rightarrow rand < DAP \end{cases} \quad \text{.....(3)}$$

6. The stability of the new positions is evaluated, that is, each crow checks its new position.

7. A fitness function is found for each crow for its new position.

8. Each crow's memory is updated using the following equation:

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

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 position that was remembered, i.e. according to the value of the memory position it has.

9. Steps 5 to 8 are repeated until the maximum number of iterations is reached and the best memory position is reached, i.e. the optimal solution to the optimization problem is reached.

Technium Vol. 8, pp.60-71 (2023)
ISSN: 2668-778X
www.techniumscience.com

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

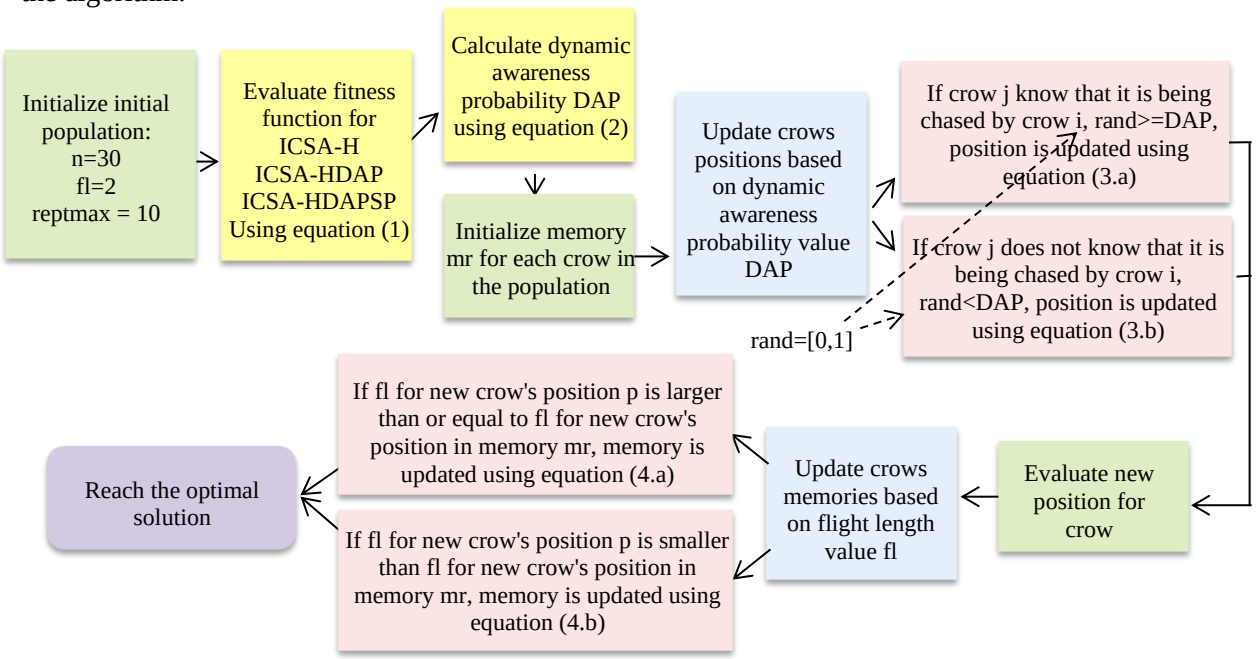

Figure 2 : Block diagram of the algorithm steps

In this research, an improvement was made on the standard algorithm to reach the desired goals and to reach the improved algorithm, where the crows positions update mechanism was improved, and this feature includes the use of the crow spiral mechanism in searching, through which the crow can follow the best position in a spiral manner and in a high dimensions area, that is, the position of the crow changes in different dimensions, and thus the best position for the crow is obtained more efficiently. The spiral search mechanism is a search space that is obtained by calculating the distance between the current position and the best position for the crow, which is represented by the following equation [20]:

$$p^{rept+1} = D \times exp(b \times l) \times cos\ cos\ (2\pi l) + gbest \quad \ldots\ldots(5)$$

where p represents the position of the crow, b represents a fixed number to determine the logarithmic form of the spiral mechanism, l represents a random number within the range [0,1], *gbest* represents the best position of the crow and D represents the distance between the best position of the crow and the current position and is calculated as follows:

$$D = |gbest - p_i^{rept}| \quad \ldots\ldots(6)$$

where p represents the position of the crow and *gbest* represents the best position of the crow.

During each iteration, the positions of some crows are randomly updated in the search space, which ensures the dispersion of these crows in space and reduces the probability of falling into the local optima. When stagnation is reached in the position update mechanism, that is, when the algorithm has occurred at the local optima, Gaussian variance and random perturbation are introduced. This method gives the crow the opportunity to escape from its current position and search in a better solution space. The Gaussian mutation is calculated as follows:

$$p^{i,rept+1} = \begin{Bmatrix} p^{i,rept} + rand & \rightarrow & r_j \geq 0.2 \\ p^{i,rept} \times Gaussian(\mu, \sigma) & \rightarrow & otherwise \end{Bmatrix} \quad \ldots\ldots(7)$$

Technium Vol. 8, pp.60-71 (2023)
ISSN: 2668-778X
www.techniumscience.com

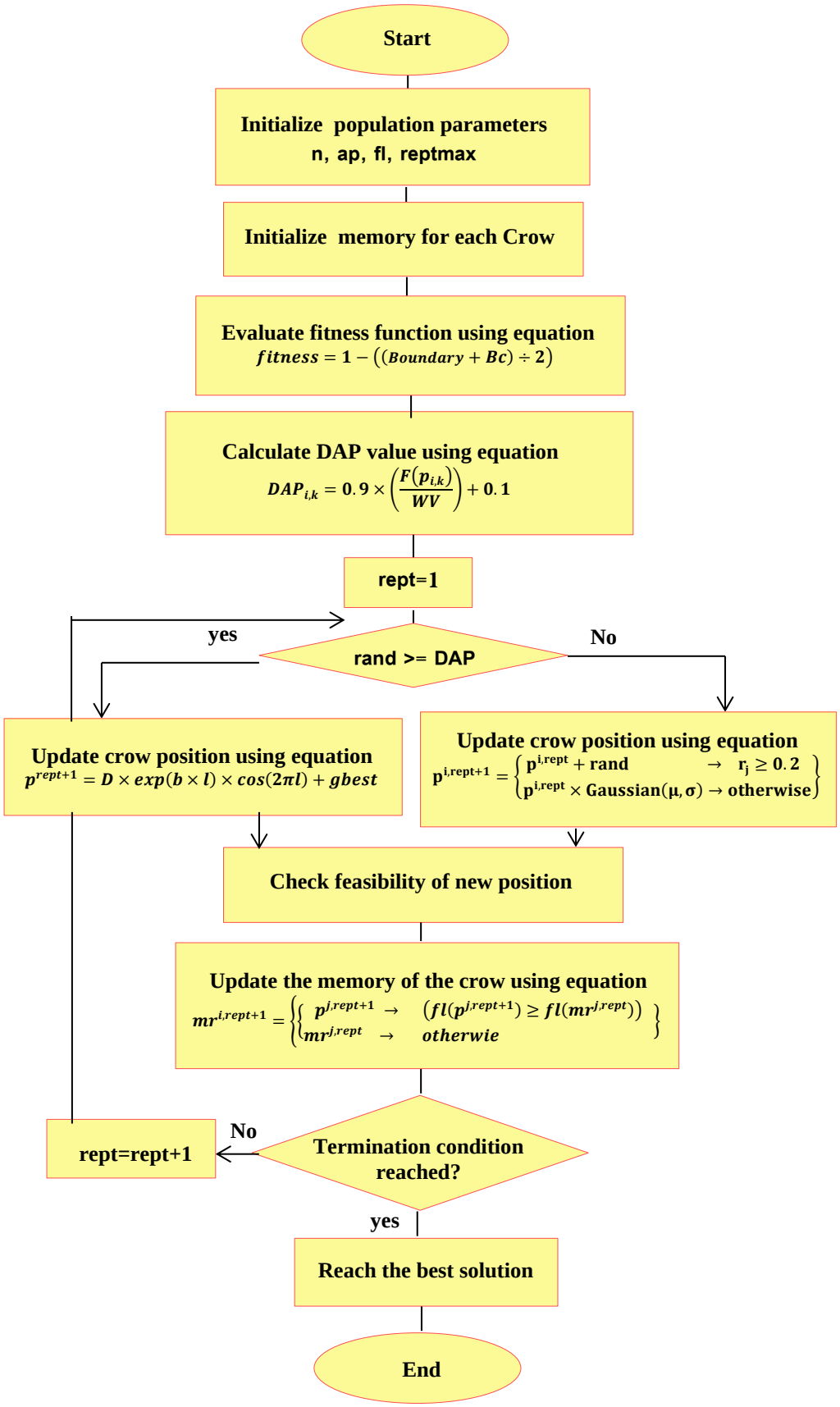

Figure 3 : A flowchart of the Method

where *p* represents the position of the crow, *rand* represents a random number within the range [0,1], *rj* represents the probability of choosing a Gaussian mutation or random perturbation and *Gaussian* represents the Gaussian variance which is calculated as follows:

$$Gaussian(\mu,\sigma) = \left(\frac{1}{\sqrt{2\pi\sigma}}\right) \times exp\left(\frac{-(p-\mu)^2}{2\sigma^2}\right) \quad \ldots\ldots(8)$$

Where μ represents the mean value, σ^2 represents the variance and *p* represents the position of the crow.

The genetic algorithm was used to prioritize the test cases obtained from the selection process using the proposed crow search algorithm. Priority is set based on a specific order in order to reach the lowest cost in terms of time, effort and budget required in the regression testing process.

The genetic algorithm helps to find the optimal arrangement of test cases, where it uses the genetic information of the chromosomes 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 its basic elements of initial population generation, chromosome representation, selection and crossover processes.

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 positions for 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 improved fitness function that combines the criteria of close to boundary value and branch coverage, and then two members of the generation are chosen to be parents in the next generation, and 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 generation, 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 generation and their ranking is approved and stored as a result of the process of prioritization.

7. RESULTS

Several programs were used to test the proposed system, and these programs include most of the components of the basic structure of the source code, namely CST1, CST2, CST3, CST4, CST5, CST6, CST7, and CST8. Work was done to reduce the test cases for the regression testing, so updated versions of the programs were used to clarify the work, as the CST2 program in the second case study is the updated version of the CST1 program in the first case study, CST4 program in the fourth case study is the updated version of CST3 program in the third case study , CST6 program in the sixth case study is the updated version of CST5 program in the fifth case study , and CST8 program in the eighth case study is the updated version of CST7 program in the seventh case study. The updated versions are programs with a new structure because the regression testing depends mainly on the state of the program after it has been subjected to the maintenance process, which includes an update or modification to the program, whether it is improvement or adaptive.

Table 1 shows the results of the ICSA-H technique, the ICSA-HDAP technique, and the ICSA-HDAPSP technique for 10 iterations, as follows:

Technium Vol. 8, pp.60-71 (2023)
ISSN: 2668-778X
www.techniumscience.com

Table 1 : 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	CST1	108.2125	120	20	16
	CST2	97.1139	90	21	15
	CST3	67.4981	90	20	15
	CST4	95.7452	120	20	14
	CST5	41.0606	90	18	11
	CST6	43.5620	90	15	12
	CST7	49.6960	240	20	14
	CST8	62.5798	240	21	17
ICSA-HDAP	CST1	99.4880	120	26	24
	CST2	90.2889	90	25	24
	CST3	59.9429	90	24	20
	CST4	90.9233	120	25	22
	CST5	40.4837	90	26	23
	CST6	40.6357	90	24	21
	CST7	37.8030	240	24	20
	CST8	57.1542	240	25	21
ICSA-HDAPSP	CST1	89.5355	120	30	30
	CST2	76.7515	90	30	30
	CST3	49.7772	90	30	30
	CST4	82.1856	120	29	29
	CST5	32.8317	90	29	29
	CST6	37.3927	90	29	29
	CST7	37.6132	240	28	28
	CST8	47.6120	240	27	27

It was observed in the ICSA-H technique that when improving the work using the improved fitness function, good results were obtained, but its work was improved by using the dynamic awareness probability value according to Equation No. (6). This was not satisfied, but its work was improved by using the spiral search mechanism of the particle to improve the work and obtain the best results with the same number of iterations of 10. The resulting test cases were all achieving the paths of the study case programs used here.

The regression testing process was carried out for the case studies that were used in this work, and the techniques used were implemented for 3 iterations. Table 2 shows the results of the regression testing of the ICSA-H, ICSA-HDAP and ICSA-HDAPSP techniques.

Technium Vol. 8, pp.60-71 (2023)
ISSN: 2668-778X
www.techniumscience.com

Table 2 : Regression testing results for ICSA-H, ICSA-HDAP, and ICSA-HDAPSP methods

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	CST2	3.6245	64	16	16
	CST4	4.3074	45	15	15
	CST6	4.0247	33	11	11
	CST8	2.6041	448	14	14
ICSA-HDAP	CST2	3.3684	96	24	24
	CST4	4.2251	60	20	20
	CST6	3.3041	69	23	23
	CST8	2.4574	640	20	20
ICSA-HDAPSP	CST2	3.0451	120	30	30
	CST4	3.6497	90	30	30
	CST6	2.1463	87	29	29
	CST8	2.0478	896	28	28

From observing the tables for all the above techniques, it was noted that the implementation of the regression testing took a little time due to the benefit of the test cases that were selected in the selection process, as the second case study relied on the data of the first case study, the fourth case study relied on the data of the third case study, and the sixth case study relied on the data of the fifth case study and the eighth case study relied on the data of the seventh case study. When the number of variables was increased when the improvement was made to the program, the data of the variables were randomly repeated in the modified version based on the previous version of the program. It was also noted that all the test cases that were selected achieve all the basic paths of the programs, and that the number of test cases that were selected increased when implementing the improved techniques, that is, by using the hybrid fitness function, relying on the crow's awareness probability value, as well as when adopting the crow's spiral search mechanism.

After the selection process has been performed and test suites have been obtained using the crow search algorithm, the genetic algorithm is used to perform the process of assigning precedence to those cases so that the best test cases are obtained in a certain order, when the algorithm is executed on CST1, 12 test cases were obtained, in CST2, 9 test cases were obtained, in CST3, 6 test cases were obtained, in CST4, 16 test cases were obtained, in CST5, 27 test cases were obtained, in CST6, 6 test cases were obtained, in CST7, 16 were obtained and in CST8, 16 test cases were obtained.

8. 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 sufficiently. It gave a small number of test cases, and a number of them did not achieve all the basic paths of the program. Therefore, it was proposed to make modifications to improve the performance of the algorithm and obtain the best results, when improving the fitness function and relying the improved fitness function, calculating the awareness probability value and adopting the spiral search mechanism for crows, very satisfactory results were obtained. In the CST3 program, during ten iterations, 30 test suites were selected out of 90 test cases, and all of them achieved the basic paths of

the program, and in 49.77 seconds after it had taken 59.94 seconds to find 24 test suites, and 20 of them achieve the basic paths of the program before improving the spiral search mechanism and it took 67.49 seconds to find 20 test suites and 15 of them achieve the basic program paths.

REFERENCES

[1] 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.

[2] Ateşoğulları, D., & Mishra, A. , (2019) , "White Box Test Tools: A Comparative View" , *International Journal on Information Technologies & Security*, *3*, p. 79-90.

[3] Lonetti, F., & Marchetti, E. , (2018) , "Emerging software testing technologies" , *Advances in computers* , Vol. 108, p. 91-143, Elsevier.

[4] Skandan P. S., Vishal R.,& Vikas R. S, (2021), "An Overview of Artificial Intelligence, Machine Learning, Internet of Things, Blockchain and Big Data", *International Journal of Creative Research Thoughts (IJCRT*), Vol.9, No.11 , pp. 328-338.

[5] Qasim, M., Bibi, A., Hussain, S. J., Jhanjhi, N. Z., Humayun, M., & Sama, N. U. , (2021) , "Test case prioritization techniques in software regression testing: An overview", *International Journal of Advanced and Applied Sciences*, 8(5) , P. 107-121.

[6] Ghai, S., & Kaur, S. , (2017) , "A Hill-Climbing Approach for Test Case Prioritization" , *Int. J. Softw. Eng. Its Appl*, 11(3), p. 13-20.

[7] Jamil, M. A., Arif, M., Abubakar, N. S. A., & Ahmad, A. , (2016) , "Software testing techniques: A literature review" , *2016 6th international conference on information and communication technology for the Muslim world (ICT4M)*, p. 177-182, IEEE.

[8] Alkawaz, M. H., & Silvarajoo, A. , (2019) , "A survey on test case prioritization and optimization techniques in software regression testing" , *2019 IEEE 7th Conference on Systems, Process and Control (ICSPC)* , pp. 59-64 , IEEE.

[9] Sharma, S. N., & Sehgal, N. , (2018) , "Enhanced test case prioritization technique using bat algorithm", *IJARIIT*, *4*(2), p. 1424-1428.

[10] Suman, S. , (2012), "A genetic algorithm for regression test sequence optimization", *International Journal of Advanced Research in Computer and Communication Engineering*, *1*(7), p. 478-481.

[11] Rehan, M., Senan, N., Aamir, M., Samad, A., Husnain, M., Ibrahim, N., ... & Khatak, H. , (2021) , "A Systematic Analysis of Regression Test Case Selection: A Multi-Criteria-Based Approach", *Security and Communication Networks*.

[12] 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.

[13] Ansari, A., Khan, A., Khan, A., & Mukadam, K. , (2016) , "Optimized regression test using test case prioritization" , *Procedia Computer Science*, *79*, p. 152-160.

[14] Kaur, A., & Agrawal, A. P. , (2017) , "A comparative study of bat and cuckoo search algorithm for regression test case selection" , *2017 7th International Conference on Cloud Computing, Data Science & Engineering-Confluence* , p. 164-170 , IEEE.

[15] Dhareula, A., & Ganpati, P., (2019), "Cuckoo Search Algorithm for Test Case Prioritization in Regression Testing", *International Journal of Recent Technology and Engineering*, *8*(3), 6004-6009.

[16] Khatibsyarbini, M., Isa, M. A., Jawawi, D. N., Hamed, H. N. A., & Suffian, M. D. M. , (2019) , "Test case prioritization using firefly algorithm for software testing" , *IEEE access*, *7*, p. 132360-132373.

[17] Vats R., Kumar A., (2020), " Test Case Prioritization Using Cat Swarm Optimization", International Journal Of Advanced Trends In Computer Science And Engineering, Vol. 9, No. 5, P. 8142.

[18] Sayed, G. I., Hassanien, A. E., & Azar, A. T., (2019) , "Feature selection via a novel chaotic crow search algorithm" , *Neural computing and applications*, 31(1), p. 171-188.

[19] 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.

[20] Han, X., Xu, Q., Yue, L., Dong, Y., Xie, G., & Xu, X. , (2020), "An improved crow search algorithm based on spiral search mechanism for solving numerical and engineering optimization problems" , *IEEE Access*, *8*, p. 92363-92382.