*International Journal of Recent Research and Review, Vol. XII, Issue 2, June 2019*
ISSN 2277 – 8322

# Proposed Binding Between Genetic Algorithm and Grey Wolf Optimizer to Estimate The Parameters of Software Reliability Growth Models

Jamal Salahaldeen Alneamy[1], Marwah Marwan AbdulazeezDabdoob[2]
[1]*Software Engineering Department, University of Mosul, Iraq.*
[2]*Software Engineering Department, University of Mosul, Iraq.*

***Abstract* - Software reliability is the main property of software quality, so make sure that the software is free of defects that cause failure. Software reliability growth models tell us the reliability of the program but first we must estimate model parameters by using optimization algorithms. In this research, a proposed binding between the genetic algorithm and the grey wolf optimization algorithm was used to estimate the parameters Of Software Reliability Growth Models.The results showed that our proposed binding (we will call it overlapping GWO_RGA) outperformed the past binding ( namely, HGWO) in parameters estimating accuracy and performance using same datasets.**

***Keywords* - Software Reliability Growth Models, grey wolf optimization algorithm.**

## I. INTRODUCTION

There is a constant need for high-quality software, and software reliability is the most measurable feature of software quality. It defines software reliability as "the probability that a program will run for a certain period in a specific environment without any failure." Defects that cause program failures are detected and removed by testing that program [2]. The software reliability growth models (SRGMs) are used to assess reliability and defined as "the mathematical relationship between the time taken in software testing and the cumulative number of detected failures" . Since 1970, many SRGMs have been introduced, which have helped software engineers measure software reliability and diagram [3]. In recent years, the meta- heuristics algorithms have gained popularity in solving the optimization problem. Therefore, in this paper, we will bind between two meta- heuristics algorithms: Grey Wolf Optimizer and Real Coded Genetic Algorithm (RGA) to highlight the advantages of each algorithm and improve the performance of estimations. The rest of this paper is organized as follows: Section 2 surveys various types of SRGMs. In Section 3 and 4, GWO and RGA are explained. In section 5, the proposed (overlapping RGA_GWO) was introduced for parameters estimation. Then, the experimental results are presented and discussed in Section 6. Finally, some conclusions are given in Section 7.

## II. THE SRGMs

Over a few decades many models have provided reliable software to estimate the reliability of the software. This reliability is modeled using a function specific to the program's failure rate according to a given statistical distribution. This program failure rate decreases gradually as the test time increases because the defect that causes the failure is corrected so the total number of defects in the program decreases. If the failure rate reaches the lowest acceptable level then reliability is achieved and the program is ready for delivery as shown in Figure 1. [4]

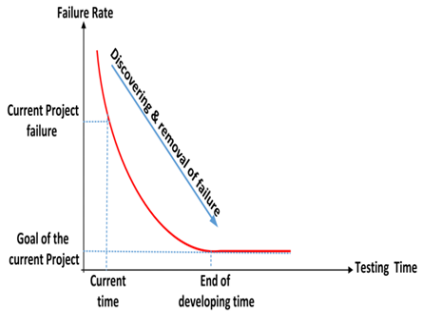


Fig.1 Software reliability modeling.

Most SRGMs have a parameter associated with the total number of defects in the code. If we knew this parameter with the current number of defects detected, we can know the number of defects remaining in the code. Knowing how many defects are left in the program code helps determine whether the program is ready for delivery and how much testing is required if the program is not ready for delivery and gives an estimation of the number of failures the customer will encounter while running the program. This estimation helps in planning the appropriate levels of support required to correct defects after delivery of the program and determine the cost of this support [5].

There are many software reliability growth models, but the most frequently used is Non Homogeneous Poisson process Model (NHPP model) which shows more accuracy than the other models [6].

NHPP models assume that the number of defects detected during time (t) follows (NHPP) with the mean value μ(t). The derivation of the function results in the average value to ʎ(t), which is the density of the failure of the means that decreases whenever defects are detected and removed. [7]

$$(t) = \frac{d\mu(t)}{d(t)} \quad \text{.....(1)}$$

There are many (NHPP) models; we will explain three of them that were used in this work, namely:

*A. Goel-Okumoto Model (G_O):*

This model was introduced by Goel and Okumoto in 1978 and is also called the Exponential NHPP Model.

Hypotheses of this model [7] :

1. All defects in the program are independent from the standpoint of failure detection.
2. The number of failures detected at any time is relative to the current number of defects in the program. This means that the probability of detecting the defect is constant.
3. Isolated defects are removed before proceeding with the test.
4. Every time a program fails, the defect that causes that failure is removed immediately and does not lead to new defects.

The G-O model assumes that the failure process is modeled using the NHPP Model using the mean value μ (t). as show in equation (2-5) and (2-6).

$$\mu(t) = a\left(1 - e^{-bt}\right) \quad \text{.....(2)}$$

$$\lambda(t) = abe^{-bt} \quad \text{.....(3)}$$

Where:

(a) Denote the initial estimate of the total failure recovered at the end of the testing process.

(b) Represents fault detection rate.

(t) Time of failure.

*B. Power Model (POW):*

It is one of the oldest models suggested by Duane in (1964). It is simple and easy to understand and the hypotheses of this model are: [8]

1. The program testing process takes place in actual operating conditions.

2 - defects are removed as soon as they are discovered.

The μ(t) and λ(t) can be given as:

$$\mu(t) = at^{b} \quad \text{..... (4)}$$

$$\lambda(t) = abte^{b-1} \quad \text{..... (5)}$$

*C. Delayed S-shaped Model (DSS):*

It was first introduced in Yamada et al. in (1983) [9].

The hypotheses of this model:

1. All defects in the program are independent from the standpoint of failure detection.

2 - The number of failures detected at any time is relative to the current number of defects in the program.

3 - Probability of detecting defect is fixed.

4. Every time a program fails, the defect that causes that failure is removed immediately and does not lead to new defects.

5. The software system is prone to failures at random times due to defects in the system.

The μ(t) and λ(t) can be given as:

$$\mu(t) = a\left(1 - (1 + bt)e^{-bt}\right) \quad \text{..... (6)}$$

$$\lambda(t) = ab^{2}te^{-bt} \quad \text{.....(7)}$$

## III. REAL CODED GENETIC ALGORITHM (RGA)

The genetic algorithm is a research technique studied by John Holland in 1970. It has the ability to solve complex optimization issues. Over the past three decades, the genetic algorithm has been used as an adaptive algorithm to solve practical problems and has been widely used in many scientific fields.

The genetic algorithm is used to find an acceptable solution (close to ideal) and shortens the time and effort required by system and software designers,

because it is a general algorithm that resolves different types of issues, taking into account the changes required by the specificity of each issue [10].

The general steps of RGA are illustrated in Figure 2:

```
Begin
Generate the initial population of chromosomes
Define fitness function f (x), x = (x1, x2, ..., xd)
Calculate fitness function of all individual chromosome
Select parents by top-mate selection
Initial probabilities of crossover (pc) and
mutation (pm)
While (t < Max Generation) or (stop criterion);
    If pc >rand
        Generate new solution by Heuristic Crossover
    End if
    If pm >rand
        Generate new solution by Non-Uniform  Mutation
    End if
    Put the solutions in the new generation
End while
End
```

Fig.2 Pseudo code of the RGA [6] .

For comparison purposes with [6] the same types of selection, crossover and mutation will be used as explained below:

_Top Mate Selection

The first father is chosen, who has the best fitness function, while the second father is chosen randomly .

_Heuristic Crossover

Used with real encoding. It depends on the value of parents' fitness to produce children. Only one child is produced. The other child is generate by passing the best-fit father to the new generation without any processing. The following equations illustrate this type of crossover:

$\text{fitness}_{\text{parent1}}$ is better than $\text{fitness}_{\text{parent2}}$ .... (8)

$\text{offspring}_2 = \text{parent}_1 \pm \text{r*}(\text{parent}_1\text{-parent}_2)$ .... (9)

Where (r) is a random value between 0 and 1.

_Nonuniform Mutation:

Used in real encoding, where the value of the parent chromosome changes in a limited range given the current generation number. If the current generation number is small, the change radius of the chromosome is large and as the generation number increases, the range of change decreases.

$$\text{offspring} = \begin{cases} \text{parent}_i + (\text{upperbound}_i - \text{parent}_i) * f(G) \\ \text{or} \\ \text{parent}_i - (\text{parent}_i - \text{lowerbound}_i) * f(G) \end{cases} \quad ..(10)$$

Where:

f(G): is the range function considering the number of the current generation (G). The function *f*(*G*) is as follows:

$$f(G) = \left(r * \left(1 - \frac{G}{G_{max}}\right)\right)^b \quad \text{.....(11)}$$

Where:

($G_{max}$): Is the maximum number of generations

(*b*): Is a shape parameter.

(*r*): Is a uniform random number between 0 and 1.

## IV. GREY WOLF OPTIMIZER (GWO)

The grey wolf algorithm was introduced by Mirjalili in 2014 and is one of the intelligence of the squadron [11]. This algorithm mimics the social leadership and behavior of phishing grey wolves in nature. Where the division of society into four sections :

alpha and denoted by (α), beta and denoted by (β), delta and symbolized by (δ), omega and denoted by (ω).

The main steps of grey wolf hunting are as follows:

1) Tracking, chasing, and approaching the victim.

2) Pursuing, encircling, and harassing the victim until it stops moving.

3) Attack towards the victim.

for modeling encircling behavior, some equations are considered:

$$\vec{D} = \left|\vec{C} \,.\, \vec{X}_p(t) - \vec{X}(t)\right| \quad \text{..... (12)}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \,.\, \vec{D} \quad \text{..... (13)}$$

Where:

(t) Indicates the current cycle.

(A) and (C) Are coefficient vectors.

($X_p$) Is the position vector of the prey.

(X) Is the position vector of a grey wolf.

The vectors A and C are calculated as follows:

$$\vec{A} = 2\vec{a} \,.\, \vec{r_1} - \vec{a} \quad \text{.....(14)}$$

$$\vec{C} = 2 \,.\, \vec{r_2} \quad \text{.....(15)}$$

Where:

(a) Is linearly decreased from 2 to 0 over the course of cycles.

( r1 and r2) are random vectors in [0, 1].

In GWO, the first three best solutions obtained are saved so far and compel the other search agents (including the omegas) to update their positions due to the position of the best search agents. The following formulas are proposed for this regard.

$$\overrightarrow{D_\alpha} = \left|\overrightarrow{C_1} . \overrightarrow{X_\alpha} - \vec{X}\right| \quad \ldots\ldots(16)$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1} . \left(\overrightarrow{D_\alpha}\right) \quad \ldots\ldots(17)$$

$$\overrightarrow{D_\beta} = \left|\overrightarrow{C_2} . \overrightarrow{X_\beta} - \vec{X}\right| \quad \ldots\ldots(18)$$

$$\overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2} . \left(\overrightarrow{D_\beta}\right) \quad \ldots\ldots(19)$$

$$\overrightarrow{D_\delta} = \left|\overrightarrow{C_3} . \overrightarrow{X_\delta} - \vec{X}\right| \quad \ldots\ldots(20)$$

$$\overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3} . \left(\overrightarrow{D_\delta}\right) \quad \ldots\ldots(21)$$

$$\vec{X}(t+1) = \frac{\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}}{3} \quad \ldots\ldots(22)$$

Alpha, beta, and delta estimate the victim position and other wolves update their positions around the victim. Pseudo code of the algorithm is shown in Figure. 3.

```
Grey Wolf Optimizer
Begin
  Initialize the grey wolf population Xi (i=1,2,….., n)
  Initialize a, A, and C
  Calculate the fitness of each search agent
  Xα= the best search agent
  Xβ= the second best search agent
  Xδ= the third best search agent
  While (t < Max number of iteration)
    For each search agent
      Update the position of current search agent
    End for
    Update a, A, and C
    Calculate the fitness of each search agent
    Update Xα, Xβ andXδ
    t=t+1
  End while
  Return Xα
End
```

Fig.3Pseudo code of the GWO [11].

## V. PROPOSED BINDING BETWEEN GWO AND RGA

The main idea of binding the swarm algorithm with the genetic algorithm is to divide the population into a first section where the grey wolf algorithm is used, and a second section where the real coded genetic algorithm is used to find the solution. When the new population is created, the real coded genetic algorithm is applied to the first section, which is the result of the grey wolf algorithm in the previous iteration. The grey wolf algorithm is also applied to the second section, which is the result of the genetic algorithm in the previous iteration, thus incorporating the solution and then re-dividing it until Stop condition is met as shown in Figure 4 and 5 .The grey wolf algorithm was considered to be the main algorithm, in which the number of gents or individuals of the population was larger than those of the genetic algorithm. This diversity and mobility between the two methods of algorithms provides a great opportunity to find the best solution and prevent falling in local optima, and this is done by updating wolfs positions solutions after selection, crossover and mutation and vice versa.


Fig.4Proposed Binding Between GWO and RGA.

## VI. TESTS AND RESULTS

### *A. Experimental Data used in this work*

Datasets used in this work are chosen in accordance to those referenced by other researchers with which the comparisons were made; the dataset(compared with HGWO) is taken from [5].

### *B. Comparison with other research*

Our proposed binding will be compared to previous binding called (HGWO) which it presented in [5] it assumed that the total numbers of iterations are equally shared by both (RGA) and (GWO). In the first step, the first half of the iterations are given to (RGA) that explores the global search place, then the solution that obtained from (RGA) is given to (GWO). In the second step (GWO) explores search space starting with the solution obtained by RGA that is set as initial population of (GWO) and continue the manipulation to find new enhanced solutions.

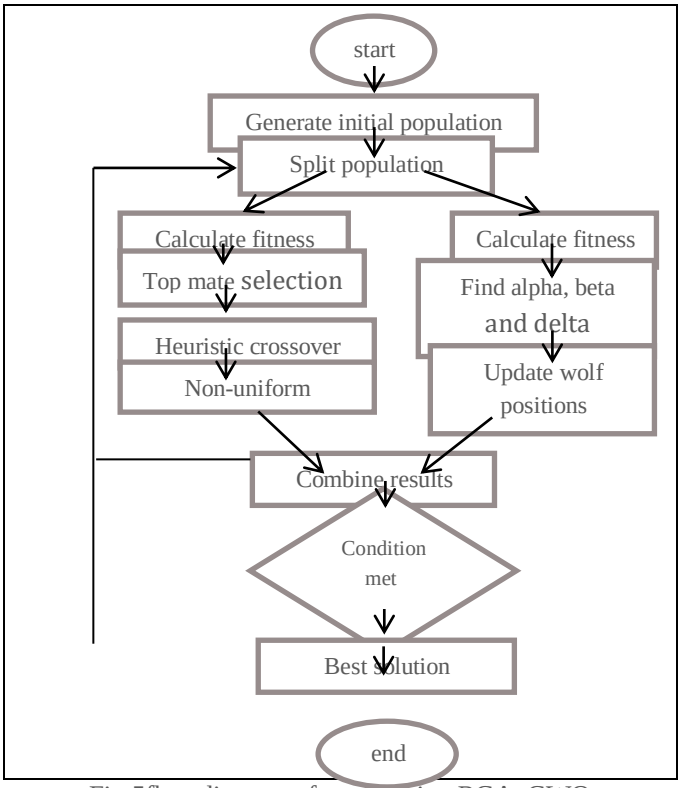


Fig.5flow diagram of overlapping RGA_GWO

The tuning parameters for the (overlapping GWO_RGA) are in Table I.

TABLE I

The tuning parameters for the overlapping GWO_RGA

| Operator | Value |
|---|---|
| Domain of search for *a* | [-1000,1000] |
| Domain of search for *b* | [-1,1] |
| Search dimensions | 2 |
| No. of search agents | 20 |
| Maximum Cycle Number | 1000 |
| Chromosome representation | Value encoding |
| Selection | Top-mate selection |
| Crossover | Heuristic Crossover |
| Mutation | Non-Uniform Mutation |
| Crossover rate | 0.5 |
| Mutation rate | 0.1 |

Three models were used: (G_O, POW and DSS). For the comparison criteria, (RMSE) is used. Results in Table II show the (RMSE) for (HGWO and overlapping GWO_RGA ), whenever the (RMSE) is less that means the best solution we have. The (overlapping GWO_RGA) outperformed the (HGWO) for all models.

In Figures 6,7 and 8 observed and estimated data were plotted.

## VII. CONCLUSION

In this work, (overlapping GWO_RGA) was used to estimate the parameters of three (SRGMs) models: G_O, POW and DSS. A comparison was made between (HGWO) and proposed (overlapping GWO_RGA), A comparison shows that our proposed algorithm present more accurate estimation and needs less iteration to reach the solution

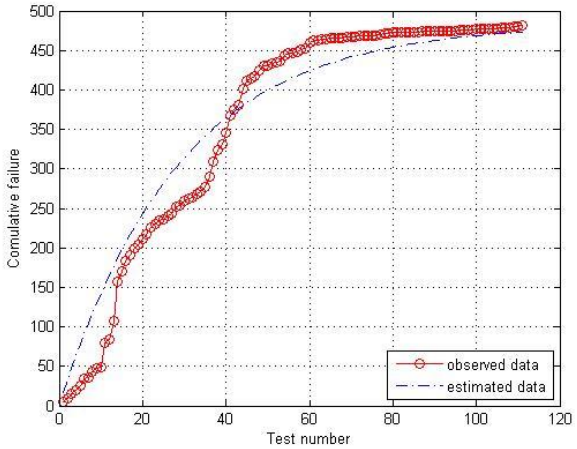


Fig.6Observed and estimated data using G_O model

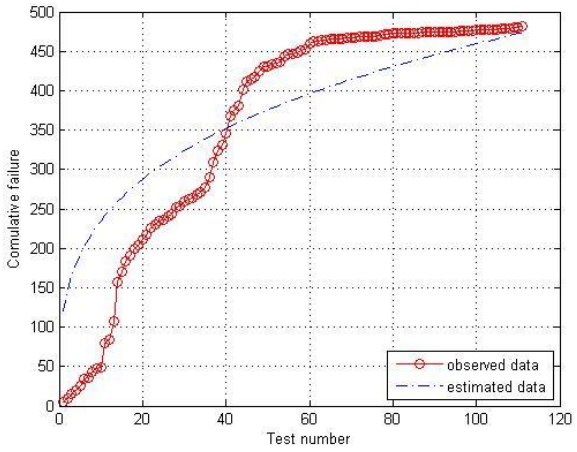


Fig.7Observed and estimated data using POW model

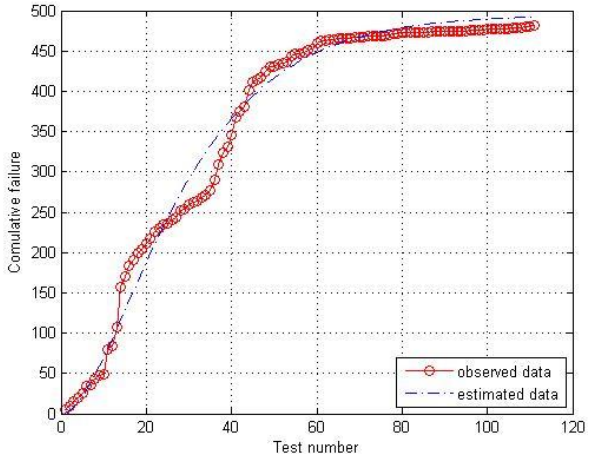


Fig.8 Observed and estimated data using DSS model

TABLE II
Comparison between HGWO and overlapping GWO_RGA

| Model | Hybrid GWO | | overlapping GWO_RGA | | | |
|---|---|---|---|---|---|---|
| | RMSE-testing | no. of iterations | RMSE-testing | no. of iterations | Parameter a | Parameter b |
| G_O | 77.893 | 516 | 7.943 | 263 | 529.568 | 0.0261 |
| POW | 146.426 | 505 | 14.812 | 248 | 83.1514 | 0.3846 |
| DSS | 16.497 | 513 | 1.1440 | 64 | 479.746 | 0.0757 |

## VIII. REFERENCES


[1] Shanmugam, L., Florence, L., 2012, "*A Comparison of Parameter Best Estimation Method for Software Reliability Models*", International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.5, pp.91-102.

[2] Kaswan, K.S., Choudhary , S., Sharma, K., 2015, "*Software Reliability Modeling using Soft Computing Techniques: Critical review*", J Inform Tech SoftwEng 5: 144.

[3] Su, Y.S., Huang, C.Y., 2006, " *Neural-network-based approaches for software reliability estimation using dynamic weighted combinational models* ", The Journal of Systems and Software 80, pp.606–615.

[4] Lyu, M. R., 1996, "*Handbook of Software Reliability Engineering*", IEEE Computer Science Press and McGraw-Hill Publishing Company, pp.1-850.

[5] Wood, A., 1996, "*Software Reliability Growth Models*", Tandem Tech., Technical Report, Vol. 96.1, Tandem Computers Inc., Corporate Information Center, Cupertino Calif., Part Number 130056.

[6] Alneamy, J. S., Dabdoob, M. M., 2017, "*The Use of Original and Hybrid Grey Wolf Optimizer in Estimating the Parameters of Software Reliability Growth Models*", International Journal of Computer

[7] Meyfroyt, P. H. A., 2012, "*Parameter Estimation for Software Reliability Models*", thesis, Eindhoven: TechnischeUniversiteit Eindhoven, pp.1-65.

[8] Kharchenko V., Tarasyuk O., Sklyar V., Dubnitsky V., 2002, " *The method of software reliability growth models choice using assumptions matrix* ",CONFERENCE PAPER.

[9] Williams, P., 2006, "*prediction capability analysis of two and three parameters software reliability growth models*", information technology journal 5(6), pp.1048-1052.

[10] AL Neamy, J. S., 2006, "*Brain Tumors Images Diagnosis Using Hybrid Intelligency Techniques*", Ph.D. Thesis, college of computers and mathematics science/university of Mosul.

[11] Mirjalili, S. A., Mirjalili , S. M., Lewis, A., 2014, "*Grey Wolf Optimizer*", Advances in Engineering Software 69, pp. 46–61.