International Journal of Recent Research and Review, Vol. IX, Issue 4, December 2016

ISSN 2277 – 8322

On the Performance of Firefly Algorithm in Software Reliability Modeling

Najla Akram AL-Saati, Marrwa Abd-AlKareem Alabajee

Software Engineering Dept., University of Mosul, Iraq

Abstract - **Reliability usually covers all parts of the system taking into account hardware, software, interfaces, operators and procedures. Whereas Software Reliability resembles a very important attribute of software quality, where the reliability of a component or system is understood and predicted prior to its implementation, this is called Reliability Modeling. Software Reliability is usually measured using software reliability growth models (SRGMs). In this paper the Firefly algorithm is evaluated for estimating the parameters of software reliability growth models. Based on real software failure data, the experiments are performed and the results are compared with Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO) and extended ACO (Ex-ACO). Experimental results show that FA is effective for estimating the parameters of SRGMs, and it has outperformed both PSO and ACO, except for some few cases where it was outperformed by Ex-ACO.**

Keywords- **Parameter Estimation, Software Reliability Growth Models, Swarm Search, Firefly Algorithm, Particle Swarm Optimization,** ACO, Ex-ACO

I. INTRODUCTION

The use of computers has spread so wide that it has become essential in everyday life activities, such as banking, industry, trade, small and even large systems like water distribution and navigation systems, etc. Computers are controlling critical applications in hospitals, controlling traffic and airplane flights, where a slightest failure can lead to a disaster and can cost human lives.

Constructing reliable software can be a very challenging problem. Issues such as Schedule Pressure, limited Resource, or vague requirements can all severely affect reliability. The development of reliable software is particularly hard when there is interdependence among software modules as noticed in much of the existing software [1].

Software reliability can define software quality, as it refers to "how well software meets its requirements" and also "the probability of failure free operation for the specified period of time in a specified environment" [2]. Several models have been established through the estimation of initial fault number and their effect on software operations, and also to predict the reliability of software [3].

A *software reliability model* can be defined as “the mathematical relation found between time consumed by software testing and the accumulative amount of errors discovered”[4].

Usually, there are two types of software reliability models [1]:

- Predicting software reliability from design parameters, this type of models is usually called "Defect Density Models”.
- Predicting software reliability from test data, this type of models is usually called "Software Reliability Growth Models".

Many Software Reliability Growth Models (SRGMs) have been proposed since 1970, and were used for estimating the reliability growth of software products. SRGMs can be employed to show the behavior of detected failures characterized of software by either times of failures or by the number of failures at fixed times [5].

In general, the parameters of SRGMs are unknown and are estimated according to the collected failure data. The most popular estimation techniques used are Maximum Likelihood Estimation (MLE) and Least Squares Estimation (LSE). Actually, MLE and LSE include the property of probability theory and statistical analysis. Thus, this could enforce certain restrictions on the process of parameter estimation for SRGMs[5].

Recent research has been encountering a massive progress regarding the application of

Evolutionary Computational techniques in Software Engineering, especially in discovering good-enough solutions to problems such as prediction, estimation and optimization [6].

In this work the Firefly Algorithm (FA), one of the Swarm Intelligent techniques, is to be used for estimating the parameters of the SRGMs. This is done with the use of real *failure data* to demonstrate the performance of the employed algorithm. Results will be compared with those obtained using three models, the Exponential (Goel-Okumoto), S-shaped and Power models, whose parameters were estimated using Particle Swarm Optimization (PSO) [7]. In addition the results are to be compared with those achieved by Ant Colony Optimization (ACO) and extended ACO [8] using the same previously mentioned models along with the M-O model.

II. RELATED WORK

Models of SRGMs were considered for studying and analysis through the literature for many years and here are some of these studies:

- 2003:Xie, Hong, and Wohlin[9], suggested a method to estimate model parameters using the available information of earlier project releases for early prediction of reliability.
- 2004: Okamura, Murayama, and Dohi[10], introduced a unified parameter estimation method established on the EM (Expectation-Maximization) principle for discrete software reliability models.
- 2005: Huang [11] presented a performance analysis of SRGMs with testing effort and change-point.
- 2006:Sheta [7]employed PSO to solve the parameter estimation problem for the exponential, power and S-Shaped models.
- 2007: Huang, Kuo, and Lyu[12], proposed new models by integrating the logistic testing-effort function into both exponential-type, and S-shaped models, parameters for the models are estimated using Maximum Likelihood Estimation (MLE).
- 2008: Hsu, Huang, and Chen [13], recommended a modified genetic algorithm with calibrating fitness functions, weighted bit mutation, and rebuilding mechanism for the parameter estimation of software reliability growth models (SRGMs).
- 2009:Yadav and Khan[14], presented taxonomy for software reliability models, the models under review reflect either infinite or finite number of failures. All exponential distribution models reflect finite failures. In contrast, logarithmic distribution based model reflect infinite failures.
- 2010: Satya Prasad, Naga Raju, and Kantam [15], proposed anew model that combine both imperfect debugging and change-point problems into SRGM.
- 2011: Gupta, Choudhary, and Saxena [16], made an analysis for software reliability using Yamada S-shaped model and generalized it by including imperfect debugging and time delay function.
- 2012: Shanmugam and Florence[3] made a comparison among parameter best estimation methods and pointed that that ACO was the best. In the same year, they made an improvement [8] on ACO and compared it to their previous work.
- 2013: Anjum, Haque, and Ahmad [17], developed a computational methodology based on weighted criteria to analyze the performance for various NHPP models. Also in 2013, Al-Saati and Alabaje [6] investigated the use of Cuckoo Search in estimating the parameters for a number of SRGMs.
- 2014: Srinivasa Rao [18], proposed software prediction models based on software reliability to improve the failure data, it was considered as a Non-Homogeneous based exponential distribution.
- 2015: Kaur[19] used a CASRE tool to measure the reliability of software using some of the models. Also that year, Wayne and Modarres [20] presented a new method for projecting the reliability growth of a complex continuously operating system. The model tolerates arbitrary corrective action strategies.

III. SOFTWARE RELIABILITY GROWTH MODELS (SRGMs)

A. Definitions of SRGMs

Throughout the last passing decades, various software reliability growth models (SRGM) have been established to be used for evaluating the status of development in the course of testing [21].

Software reliability growth models (SRGM) define the overall form of the dependence of the failure process on the basic factors that affect it:

- Fault Introduction,
- Fault Removal, and
- The Operational Environment.

In general, failure rates of software system decreases with time after being forced by both Fault Identification and Removal. The main purpose of a software reliability model is to predict failure actions expected to be experienced when the software is in operation. Such a predicted behavior varies quickly and can be observed throughout the test period of the program [22].As faults are detected and repaired, Models of SRGM become important in estimating the enhancement of software reliability [23].

Software reliability modeling includes mathematical and statistical functions that employ quite a few computational steps. The parameters of the Models' Equations are estimated with techniques such as least squares fit or maximum likelihood estimation [24]. The fact is that each model is able to provide good result for a certain data set; however no model can provide such results for all data sets [17].

In SRGM, reliability can grow through testing time *t*, which is typically measured in terms of the CPU execution time in use, or the amount of man-hours or days. This growth is commonly identified either in terms of failure intensity$\lambda(t)$, or in terms of the mean value function $\mu(t)$[25]. A number of parameters must be estimated by the failure data collected to calculate the mean value function [9].In general SRGMs encounter some major challenges [2]:

- Software testers don't usually follow the operational profile in testing the software, thus what is seen through software testing may not be straightly extensible for operational use.
- Once the number of failures collected is limited, it becomes difficult to carry out statistically expressive reliability predictions.
- Some assumptions of SRGM are not realistic.

Terms used in defining the models in the next subsections are given in TABLE I.

TABLE I
Terms used for defining SRGMs

Term	Definition
$\mu(t)$	Denotes the mean failure function, i.e., the expected number of failures observed over a period of time t.
$\lambda(t)$	Is the failure intensity function, i.e., failure rate
a	The initial estimate of the total failure recovered at the end of the testing process.
b	Represents the ratio between the initial failure intensity $\lambda 0$ and total failure.
NHPP	The Non Homogenous Poisson Process: provides probability that the number of failures at a time t will have a particular value.

B. Classification of SRGMs

SRGMs can be classified in to two types [26]:

- Models described in terms of the failure times of the process, most popular known class models of this type is the class of the General Order Statistics (GOS) models , it is assumed that the unknown initial number of faults (N) in a software system be unknown but fixed.
- Models described in terms of the number of observed failures, the most popular class models of this type, is the class of non-homogeneous Poisson process (NHPP) models. It is assumed that the unknown initial number of faults (N) in a software system be a random variable following a Poisson distribution.

In this work, models from the second type are used.

C. General NHPP Model

One of the most commonly studied SRGMs is the Non-Homogeneous Poisson Process (NHPP), and because of its mathematical tractability and wide applicability, it became one of the main classes of the present SRGMs [27]. NHPP models are also called *fault counting models* which can either be finite or infinite models, according o how they are stated [24].

NHPP models are considered to be very advantageous in the analysis of reliability, particularly in repairable systems. These models can be used for both software and hardware, and for combined systems, this is mainly due to the fact that hardware systems are usually repairable, and software debugging is also a repair process [28].

The NHPP based models are considered to be simple, convenient and compatible. It is essential, in these models, to specify a suitable mean value function to represent the probable number of failures occurred up to a precise time point. Being NHPP based SRGMs; they can be successful in practical software reliability engineering [29].The number of detected failures up to time (t) can be modeled as a pure birth counting process $\left(\mathrm{N}(\mathrm{t})\right)_{t\geq 0}$[4].

It is assumed that the number of software failures during non-overlapping time intervals don't have any impact each other. That means, for any finite collection of times $t_1 < t_2 < \cdots < t_n$ the "n" random variables $\{N(t_2) - N(t_1)\}, \ldots. \{N(t_n) - N(t_{n-1})\}$ are independent. Therefore, the counting process {N (t), t>0} has independent increments [29].
If the expected number of software failuresis represented by $\mu(t)$in time (t), then the mean value function $\mu(t)$ is finite valued, non-decreasing, non-negative and restricted with the boundary conditions. Assuming$N(t)$ is known to have a Poisson probability mass function with parameters $\mu(t)$ as in Eq.(1), then N(t) is called NHPP. Hence the stochastic behavior of software failure phenomena can be described through the N(t) process [29].

$$P[N(t) = n] = e^{-(\mu(t))} \frac{(\mu(t))^n}{n!} \qquad \ldots\ldots\ldots\ldots\ldots\ldots(1)$$
Where n=0, 1, 2, …,∞

D. Basic Assumptions of NHPP-Models

The followings are some of the basic assumptions for NHPP models [24]:

- A Software system is subject to failure during execution caused by faults remaining in the system.
- The number of faults detected at any time is proportional to the remaining number of faults in the software.
- Failure rate of the software is equally affected by faults remaining in the software.
- On a failure, repair efforts starts and fault causing failure is removed with certainty.
- All faults are mutually independent from a failure detection point of view.

E. NHPP-Models Used in this Work

A number of models is considered in this work, mainly those that are used more often and commonly in the literature. TABLE II shows the employed models, refer to [6] for more details.

TABLE II
Employed NHPP Models

SRGMs	$\mu(t)$	$\lambda(t)$
Exponential Model (Goel-Okumoto G-O)	$\mu(t) = a(1 - e^{-bt})$	abe^{-bt}
The Power Model (POW)	$\mu(t) = at^b$	$abte^{b-1}$
Yamada Delayed S-Shaped Model (DSS)	$\mu(t) = a(1 - (1 + bt)e^{-bt})$	ab^2te^{-bt}
Musa-Okumoto Logarithmic Model (M-O)	$\mu(t) = a * \ln(1 + bt)$	$\frac{ab}{(1 + bt)}$

IV. SWARM INTELLIGENCE

A. Firefly Algorithm

Firefly Algorithm(FA) is one of the swarm intelligence optimization methods proposed by Xin-She Yang at University of Cambridge in 2007[30-31].FA is inspired by fireflies' behavior in nature, and although this algorithm has many similarities with other algorithms which are based on the so-called swarm intelligence, such as the famous Particle Swarm Optimization (PSO), Artificial Bee Colony optimization (ABC), and Bacterial Foraging (BFA) algorithms, it is indeed much simpler in both concept and implementation. Also, according to recent bibliography, it is very efficient and can outperform conventional algorithms like genetic algorithms for solving several optimization problems [32].

This algorithm is inspired by social behavior of fireflies and the phenomenon of Bioluminescent Communication [30]. Most of the fireflies produce short and rhythmic flashes; their flashing light is generated by a process of bioluminescence and may serve as an element of engagement rituals or warning signals [33].

B. Behavior of Fireflies

Fireflies usually release a flashing light in an amazing phenomenon and they sparkle in the atmosphere of the tropical and temperate regions. There exist about two thousand species of firefly; most of them produce short and rhythmic flashes. The pattern of flashes produced by fireflies is often unique for a particular species. A process of bioluminescence is responsible for the production of such a flashing light, and the real functions of those signaling systems are still in assessment. Nevertheless, two fundamental functions of these flashes are:

- Attracting mating partners (communication),
- Attracting potential prey.

Moreover, flashing may also function as a protective and defensive warning mechanism. The rhythmic flash, flashing rate and amount of time, all formulate part of the signal system that brings both sexes together. [34].

Females usually respond to a unique pattern of flashing of males in the same species, whereas in some species, females are able to mimic the mating flashing pattern of other species in order to bait and eat males who might mistake the flashes as a potential appropriate mate [35].

As a fact, the intensity of light at a definite distance (r) from the light source adapts to the inverse square law. To be precise, the light intensity (I) decrease as the distance (r) increases in terms of ($I \alpha 1/r^2$). Furthermore, the light keeps being absorbed by the air and becomes weaker as the distance increases. When these two factors are combined, most fireflies become visible at a limited distance, typically to a few hundred meters at night, which is fairly sufficient for fireflies to communicate with each other [36].

C. Concept

To construct a firefly-inspired algorithm, some of the flashing characteristics of fireflies have to be idealized. In accordance, these flashing characteristics can be summarized using the following three rules 36]:

- All fireflies are unisex; therefore each firefly is attracted to other fireflies irrespective of their sex.
- Attractiveness and brightness are proportional to each other, so for any two flashing fireflies, the less bright one will move towards the brighter one. Attractiveness and brightness both decrease as their distance increases. If there is no one brighter than other firefly, it will move randomly.
- The firefly's brightness is determined by the view of the objective function.

According to these three rules, the basic steps of the firefly algorithm (FA) can be summarized as the pseudo code shown in Fig. 1 [34].

```
Objective function $f(x), x = (x_1, ......, x_d)^T$
Generate initial population of fireflies $x_i$ $(i = 1,2, .....n)$
Light intensity $I_i$ at $x_i$ is determined by $f(x_i)$
Define light absorption coefficient $\gamma$
while (t <MaxGeneration)
for i = 1 : n all n fireflies
for j = 1 : i all n fireflies
if ($I_j > I_i$), Move firefly i towards j in d-dimension
end if
Attractiveness varies with distance $r$ via $exp[-\gamma r]$
Evaluate new solutions and update light intensity
end for j
end for i
Rank the fireflies and find the current best
end while
Post process results and visualization
```

Fig.1 Pseudo code of the firefly algorithm (FA**)**

D. Light Intensity and Attractiveness

The landscape of the objective function controls the brightness of a firefly. As a maximization problem, the brightness can simply be proportional to the value of the objective function [35].

Let the brightness (I) of a firefly at a specific position (x) be considered as a simple case for maximum optimization problem, it can be stated as $I(x) \alpha f(x)$. On the other hand, the attractiveness (β) is relative; it should be realized or judged by the other fireflies. This way, it will differ with the distance (r_{ij}) between firefly (i) and firefly (j). Furthermore, the intensity of light decreases with the distance from its source, it is also absorbed by the media, so the attractiveness should be allowed to vary with the degree of absorption [34].

In Firefly algorithm, the attractiveness function $\beta(r)$ of a firefly is a monotonically decreasing function as stated in Eq. (2) [32]

$$\beta(r) = \beta_0 \, e^{-\gamma r^m} \quad ,(m \geq 1) \qquad ……………(2)$$

Where,

r: is the distance between two fireflies.

β_0 :is the attractiveness at (r = 0).

γ:is the absorption coefficient controlling the decrease of the light intensity.

E. Distance

The distance concerning any two fireflies (i) and (j) at locations (x_i) and (x_j), respectively is the Cartesian distance, and can be defined as in Eq. (3) [36]

$$r_{i,j} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^{d}\left(x_{i,k} - x_{j,k}\right)^2} \quad \text{................(3)}$$

Where

$x_{i,k}$: is the kth component of the spatial coordinate (x_i) of the ith firefly,

d: is the number of dimensions, in a 2D case, the distance is given as in Eq. (4) [36]

$$r_{ij} = \sqrt{\left(x_{i1} - x_{j1}\right)^2 + \left(x_{i2} - x_{j2}\right)^2} \quad \text{............(4)}$$

F. Movement

The movement of a firefly (*i*) attracted to another more attractive firefly (*j*) is computed using the attraction Eq.(4) where (*m=2*) multiplied by difference between x_i and x_j with α being the randomization parameter. This is shown in Eq. (5)

$$x_i = x_i + \beta_0\, e^{-\gamma r_{i,j}^2} * \left(x_j - x_i\right) + \alpha * \left(rand - \frac{1}{2}\right) \quad \text{.........(5)}$$

Where,

x_i is the current position of a firefly,

α is the randomization parameter.

rand is a random number generator uniformly distributed in the range of [0, 1].

For most cases in the implementation, β_0 = 1 and α= [0, 1]. Furthermore, the randomization term can effortlessly be extended to a normal distribution N (0, 1) or any other distribution [35].

V. TESTS AND RESULTS

A. Experimental Data and Parameter Settings

To test the efficiency of the Firefly algorithm when applied to SRGMs, comparisons are made with previous results obtained using PSO (employing three models) on datasets: Data1, Data2, and Data3 [7], and also compared with results obtained using ACO, and EX-ACO (employing four models) on Musa Data sets taken from the *Data Analysis Center* for Software's Reliability Data set [37] for Project2, Project3, and Project4. TABLE III shows the settings of the parameters for the FA used in this paper.

TABLE III
Parameter Settings for Firefly algorithm

Parameter	Value
Lower and Upper bounds for (a)	[0.00001-2000]
Lower and Upper bounds for (b)	[0.00001 – 1]
Number of fireflies (n)	25
number of dimensions (d)	2
the maximum number of generations	100
randomization parameter (α)	0.01
initial attractiveness (β_0)	1
absorption coefficient (γ)	1

B. Evaluation Criterion

In this work, and for comparison reasons, two type of evaluation criteria are used to measure the performance of the algorithm based models; the first is the Root Mean Square Error-RMSE given in Eq. (6). This measure is used here with (Data1, Data2, and Data3) datasets. The second measure is the Euclidean Distance-ED that was used by Shanmugam and Florence [8], its formulation is shown in Eq. (7) and is used with (Project2, Project3, and Project4) datasets.

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(m_i - \mu_i)^2} \quad \text{............ (6)}$$

Where,

N: is the number of measurements used for estimating model parameters,

m_i:is the actual failure number.

μ_i: is the predicted failure number.

$$ED = \sqrt{\sum_{i=1}^{N}(m_i - \mu_i)^2} \quad \text{...........(7)}$$

Where,

N, m_i, μ_i is the same as in previous equation (Eq. (6)).

C. Comparisons using RMSE

The Firefly algorithm is trained and tested using (70%, 30%) training and testing percentages respectively, the same percentages were used by Sheta [7] for Data1, Data2, and Data3, the results are then compared with those obtained using PSO [7] for (EXP, POW and DSS) Models. TABLES IV, V and VI show the results of FA and the comparisons with PSO for Data1, Data2 and Data3 using RMSE and testing 30%.Results show that FA outperformed PSO for all models in Data1, Data2, and Data3. Best results are shown in bold.

TABLE IV
Comparison of FA and PSO using RMSE (Data1)

	FA	PSO
EXP(G-O)	**15.9041**	119.4374
POW	**43.0197**	152.9372
DSS	**16.2004**	26.3015

TABLE V
Comparison of FA and PSO using RMSE (Data2)

	FA	PSO
EXP(G-O)	**22.9082**	80.8963
POW	**81.5982**	149.9684
DSS	**6.9173**	17.0638

TABLE VI
Comparison ofFA and PSO using RMSE (Data3)

	FA	PSO
EXP(G-O)	**10.7637**	13.6094
POW	**12.6660**	14.0524
DSS	**11.8653**	47.4036

D. *Comparisons using Euclidean Distance*

FA was also trained using other data sets and other training percentages, the results were compared with those achieved using ACO and extended ACO [8]which employed the same datasets and (100%) of data for each set for training for the (G-O, POW, DS S, and M-O) models for Project2, Project3, and Project4. The comparison was done using Euclidean Distance. TABLES VII,VIII and IX show the comparison among FA, ACO, and Ex-ACO.

Results for Project2 in TABLE VII show that Ex-ACO outperformed both FA and ACO for all models. Results are also depicted by Fig .2.

TABLE VII
Comparison among FA, ACO and Ex-ACO (Project2)

	FA	ACO	Ex-ACO
EXP(G-O)	42.7901	60.0371	**28.5891**
POW	46.3033	52.8854	**34.0521**
DSS	42.5206	52.8854	**33.0461**
M-O	42.2256	26.0385	**17.359**

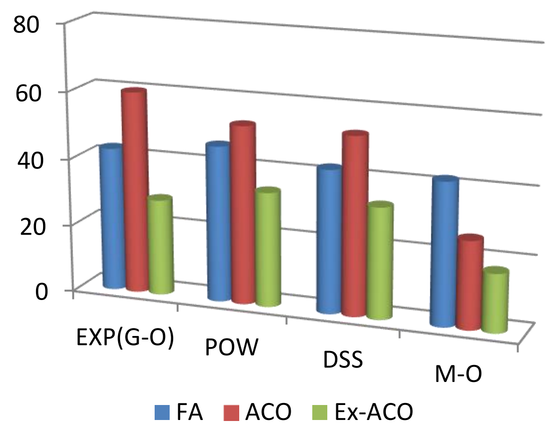

Fig. 2 Differences among Search Algorithms (Project2)

Results of Project3 in TABLE VIII, on the other hand, showed that FA was better than both ACO and Ex-ACO for all models. Fig. 3 illustrates this.

TABLE VIII
Comparison among FA, ACO and Ex-ACO (Project3)

	FA	ACO	Ex-ACO
EXP(G-O)	**30.4631**	71.5489	34.0709
POW	**15.3948**	57.5801	47.5814
DSS	**30.7734**	57.5801	48.4914
M-O	**20.5183**	36.1891	24.126

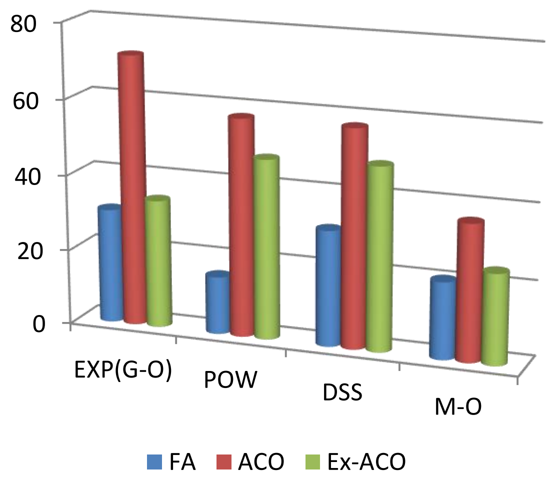

Fig .3 Differences among Search Algorithms (Project3)

Results of Project4 showed in TABLE IX and demonstrated in Fig.4 also indicated that FA performed better than all others except for the M-O model, where Ex-ACO performed better than all.

TABLE IX
Comparison among FA, ACO and Ex-ACO (Project4)

	FA	ACO	Ex-ACO
EXP(G-O)	**25.7488**	71.4015	35.0007
POW	**28.5832**	53.2234	34.2645
DSS	**25.6125**	53.2234	35.2635
M-O	26.4196	33.1728	**22.1152**

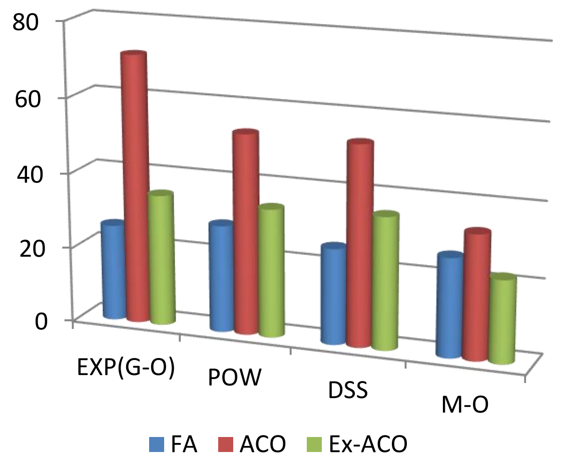

Fig. 4 Differences among Search Algorithms (Project4)

VI. CONCLUSIONS

In this work, Firefly algorithm was used to estimate the parameters of Software Reliability Growth Models. Some numerical examples based on a real failure data were used; experimental results showed that FA was effective for estimating the parameters of SRGMs. The comparison was carried out with PSO, ACO and Ex-ACO, where FA outperformed all others, apart from some cases where Ex-ACO was better.

VII. REFERENCES

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

[2] T.H. Sheakh, V. Singh, "Taxonomical Study of Software Reliability Growth Models", *International Journal of Scientific and Research Publications*, ISSN 2250-3153, Vol.2, Issue.5, pp: 1-3. 2012.

[3] L.Shanmugam, L.Florence, "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. 2012.

[4] P. H.A. Meyfroyt, "Parameter Estimation for Software Reliability Models", *Thesis, Eindhoven: Technische Universiteit Eindhoven*, pp: 1-65. 2012.

[5] C.J. Hsu, and C.Y. Huang, "A Study on the Applicability of Modified Genetic Algorithms for the Parameter Estimation of Software Reliability Modeling", *IEEE 34th Annual Computer Software and Applications Conference* 2010.

[6] N.A.AL-Saati,M. A.Alabaje, "The Use of Cuckoo Search in Estimating the Parameters of Software Reliability Growth Models",*International Journal of Computer Science and Information Security(IJCSIS)*, Vol.11, No.6, June 2013.

[7] A.Sheta, "Reliability Growth Modeling for Software Fault Detection Using Particle Swarm Optimization", *IEEE Congress on Evolutionary Computation Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada 2006*, pp: 3071- 3078.

[8] L.Shanmugam, L.Florence, "An Improved ACO Technique for Parameter Estimation", *European Journal of Scientific Research*, ISSN 1450-216X , Vol. 89, No.1, pp:101-108. 2012.

[9] M.Xie, G.Y.Hong,C.Wohlin,"Modeling and Analysis of Software System Reliability", *In Case Studies on Reliability and Maintenance*, Published by John Wiley and Sons, Inc., Hoboken, New Jersey, Online ISBN: 9780471393009, pp: 233–250. 2003

[10] H.Okamura, A. Murayama,T.Dohi, "EM algorithm for Discrete Software Reliability Models: A Unified Parameter Estimation Method", *Proceedings of theEighth IEEE International Symposium on High Assurance Systems Engineering (HASE'04)*,2004. pp: 219 - 228.

[11] C.Y.Huang, "Performance Analysis of Software Reliability Growth Models with Testing Effort and Change-Point", *Journal of Systems and Software*, Vol. 76, pp: 181-194.2005.

[12] C.Y.Huang, S.Y.Kuo, M.R.Lyu, "An Assessment of Testing-Effort Dependent Software Reliability Growth Models", *IEEE Transactions on Reliability*, Vol.56, No.2, pp: 198-211. 2007.

[13] Hsu, C.J., Huang, C.Y., Chen, T.Y., (2008), "Fast Abstract: A Modified Genetic Algorithm for Parameter Estimation of Software Reliability Growth Models", *19th International Symposium on Software Reliability Engineering*, pp: 281-282.

[14] A. Yadav, R.A.Khan, "Critical Review on Software Reliability Models", *International Journal of Recent Trends in Engineering*, Vol.2, No.3,pp:114-116.2009.

[15] R.SatyaPrasad, O.NagaRaju, R.R.LKantam,"SRGM with Imperfect Debugging by Genetic Algorithms",

I. J. of Software Engineering & Applications (IJSEA), Vol.1, No.2, pp: 66-79. 2010.

[16] A. Gupta, D. Choudhary, S. Saxena, "Software Reliability Estimation using Yamada Delayed S - Shaped Model under Imperfect Debugging and Time Lag", *International J. of Computer Applications* (0975 – 8887) Vol.23, No.7, pp: 49-52. 2011.

[17] M.Anjum, M.A.Haque, N. Ahmad, "Analysis and Ranking of Software Reliability Models Based on Weighted Criteria Value", *I.J. of Information Technology and Computer Science*, pp:1-14. 2013.

[18] P.SrinivasaRao, "Innovation of Reliability to Improve Software Predictive Models", *in International Journal of Emerging Technology and Advanced Engineering.* Vol.4, Issue.5. 2014.

[19] A.Kaur, "Comparative Analysis of Reliability Models–Based on Uncertainty Factors", *In I. J. of Advanced Research in Computer Science and Software Engineering.* Vol.5, Issue.2. 2015.

[20] M.Wayne, M.Modarres,"A Bayesian Model for Complex System Reliability Growth Under Arbitrary Corrective Actions", *In IEEE Transaction on Reliability,.* Vol.64, Issue.1. 2015.

[21] P.K.Kapur, H. Pham, S.Anand, K.Yadav, " A Unified Approach for Developing Software Reliability Growth Models in the Presence of Imperfect Debugging and Error Generation", *IEEE Transactions on Reliability,* Vol. 60, No. 1,pp:331-340. 2011.

[22] P. Nagar, B.Ankachan, "Application of Goel-Okumoto Model in Software Reliability Measurement",*IJCA Special Issue on Issues and Challenges in Networking, Intelligence and Computing Technologies ICNICT*, Published by Foundation of Computer Science, NY, pp:1-3.2012.

[23] M.Razeef, M.Nazir, "Software Reliability Growth Models: Overview and Applications", *J. of Emerging Trends in Computing and Information Sciences,* Vol.3, No.9, pp: 1309-1320. 2012.

[24] R.Lai, M.Garg, "A Detailed Study of NHPP Software Reliability Models", *Journal of Software*, Vol.7, No.6, pp: 1296-1306. 2012.

[25] Malaiya, Y.K.,(2003), "Software Reliability", Encyclopedia of Library and Information Science, Marcel Dekker, pp: 2688 - 2698.

[26] I.C.Ramos, "Statistical Procedures for Certification of Software Systems", *Netherlands Organization for Scientific Research (NWO) project no.* 617.023.047. 195p. 2009.

[27] O.Shatnawi, "Discrete Time NHPP Models for Software Reliability Growth Phenomenon", *In International Arab Journal of Information Technology*, Vol. 6, No. 2, pp: 124-131. 2009

[28] M. Xie, Y.S. Dai, K.L. Poh, "Computing System Reliability Models and Analysis", Kluwer Academic/Plenum Publishers2004.P. 293.

[29] G.R .Neppala, R.S. Prasad, R.R.L.Kantam, "Software Reliability Growth Model using Interval Domain Data", *I.J. of Computer Applications (0975 – 8887)* , Vol.34, No.9, pp:5-8.2011.

[30] G.Wang, L.Guo, H.Dua, L. Liu, H. Wang, "A Modified Firefly Algorithm for UCAV Path Planning", *I.J. of Hybrid Information Technology,* Vol.5, No.3, pp:123-144. 2012.

[31] X.S.Yang," Nature-Inspired Metaheuristic Algorithms," 2nd Edition, Luniver Press, P.160. 2010.

[32] T.Apostolopoulos, A.Vlachos, "Application of the Firefly Algorithm for Solving the Economic Emissions Load Dispatch Problem", *Hindawi Publishing Corporation International Journal of Combinatorics.* Vol.2011, pp:1-23. 2010.

[33] J.Kwiecień, B.Filipowicz, "Firefly Algorithm in Optimization of Queuing Systems", *Bulletin of the Polish Academy of Sciences Technical Sciences*, Vol.60, No.2, pp: 363-368. 2012.

[34] X.S.Yang, "Firefly Algorithms for Multimodal Optimization", *Stochastic Algorithms: Foundations and Applications, SAGA, Lecture Notes in Computer Sciences*, Vol.5792, pp:169-178. 2009.

[35] N.Chai-ead, P.Aungkulanon, P.Luangpaiboon, "Bees and Firefly Algorithms for Noisy Non-Linear Optimisation Problems", *Proceedings of the International MultiConference of Engineers andComputer Scientists*, Vol.2, pp: 1449–1454.2011.

[36] S.K.Pal, C.S.Rai, A.P.Singh, "Comparative Study of Firefly Algorithm and Particle Swarm Optimization for Noisy Non- Linear Optimization Problems", *I.J. Intelligent Systems and Applications*, pp:50-57. 2012.

[37] https://sw.thecsiac.com/databases/sled/swrel.php.