International Journal of Computer Applications (0975 – 8887)
Volume 145 – No.5, July 2016

A New Method to Optimize the Reliability of Software Reliability Growth Models using Modified Genetic Swarm Optimization

Mallikharjuna Rao K.
Assistant Professor, Department of IT,
GITAM University, Visakhapatnam,
Andhra Pradesh, India

K. Anuradha, PhD
Professor, Department of CSE,
School of Computing, GRIET, Hyderabad,
Telangana, India

ABSTRACT

Software reliability is one of the key attributes to determine the quality of a software system. Finding and minimizing the remaining faults in software systems is a challenging task. Software reliability growth model (SRGM) with testing-effort function (TEF) is very helpful for software developers and has been widely accepted and applied. However, each SRGM with TEF (SRGMTEF) contains some undetermined parameters. Optimization of these parameters is a necessary task. Generally, these parameters are estimated by the Least Square Estimation (LSE) or the Maximum Likelihood Estimation (MLE). However, the software failure data may not satisfy such a distribution. We investigate the improvement and application of a swarm intelligent optimization algorithm, namely Modified Genetic Swarm Optimization algorithm, to optimize these parameters of SRGMTEF. The performance of the proposed SRGMTEF model with optimized parameters is also compared with other existing models Genetic Algorithm (GA), and Particle Swarm Optimization (PSO). The experiment results show that the proposed parameter optimization approach using Modified Genetic Swarm Optimization is very effective and flexible, and the better software reliability growth performance can be obtained based on SRGMTEF on the different software failure datasets. Also, provided comparison of ten SRGMs (Like Goel-Okumoto Model, Delayed S-shaped Growth Model, Yamada Imperfect Debugging Models, Yamada Rayleigh Model, Inflection S-shaped Model…..etc).

Keywords

Software Reliability Growth Model, Testing Effort Function, Genetic Algorithm, Particle Swarm Optimization, Modified Genetic Swarm Optimization. Software Reliability

1. INTRODUCTION

Today's technological world is extremely depending on computer systems i.e. computers became a part of human life (Humans are directly or indirectly affected by computers). From the past three decades, the scope and intricacy of computer-intensive software systems have rapidly grown and the tendency will continue in the next generation undoubtedly. If you look at the world virtually all the industries like banking, automobiles, hospitals, avionics, telecommunications, oil, pharmaceuticals, and many industries depend on computers for their simple operations [1]. The computers are running by the combination of both hardware and software. The software is defined as transforming a discrete set of inputs into an isolated set of outputs during specified period.

In order to produce high-quality software, there is a need to test the software thoroughly i.e. how much effort did the testing team spent for testing the software. Estimation of software reliability involves recognizing and confiscation of software defects. Software defects play a vital role in reliability prediction. Modeling software reliability is a challenging task because when the identified defects removed from the system, it may result in new faults.

Identifying and removal of the residual faults are one of the key features in software reliability indexes. Numerous Software Reliability Growth Models (SRGMs) are proposed in the literature based on nonhomogeneous Poisson process (NHPP). The conventional SRGMs focused on recognition of faults and predicting the reliability based on the available past failure data.

In this paper, we use evolutionary algorithms to forecast the reliability values of software systems. Also, we worked on test case generation for exhaustive testing. In our proposed work first, we evaluate various parameters of SRGMs like the mean value of failures, the mean value of faults, the cumulative number of failures and reliability and then suggested the optimal selection of SRGM. Indeed, it contrasts with the existing methods. The proposed method is compared with existing optimization techniques. The results show that the use of evolutionary algorithms gives better results.

The rest of the paper is organized as follows. Section 2 presents the various researches performed in relation to our suggested work. Section 3 elucidates the plan, approach, and the advanced technique. Section 4 proves and details about the results of our suggested technique, and finally, section 5 closes our proposed method for parameter estimation of software reliability growth model.

2. RELATED WORK

Kapil Sharma et al. [2] have developed a method to envisage the reliability, ranking and selection of the software reliability growth models (SRGM) using distance based approach (DBA) method. Several SRGMs proposed for the last three decades. But there is no standard approach to select optimal SRGM. Selection of SRGM requires efficient estimation of reliability parameters which helps in determining the quality. The DBA method first recognizes the relative importance criteria of the given software application. Then it evaluates the level of standards for a group of models for optimal selection.

Because of the vibrant flora of software determining the reliability of a software system is a typical task during testing because of its vibrant flora. The process of fault recognition and removal are not same during development and operational stage. The elimination of defects during the operational phase is slower than development stage. To predict and assess the reliability during testing and operational stages Chin-Yu Huang et al. [3] developed a robust and easily deployable technique named unified theory. Integrating multi change-points i.e. software environment changes during different time intervals with the unified theory profoundly helped in an assessment of software reliability.

The software defined as a group of programs or modules. Due to the intricacy existed in software applications, and poor understanding of software requirements, it is not easy to remove the faults immediate recognition of software failure during testing. Eliminating the identified defects may introduce new faults called imperfect debugging. P. K. Kapur et al. [4] proposed two gentle outlines to predict the reliability of various software reliability growth models which are related to nonhomogeneous Poisson process in the presence of imperfect debugging. Their investigations focused on failure recognition and fault elimination during testing, when no discrepancy existed.

To deliver the optimal software Y. P. Wu et al. [5] have proposed an approach for detecting faults and correcting faults by incorporating time dependencies between them. This approach focused on the parameter estimations of the integrated model. The process of estimation of parameters in the integrated model is done using explicit likelihood estimation by considering different time delay assumptions. Also, it helps in optimal software release policies i.e. the time delay in the software release. In this work they employed numerous selves of the integrated model; predictive capability is evaluated and compared with the traditional Maximum likelihood and least squares estimation methods.

Genetic Algorithm [6] is one of nature inspired algorithm. To achieve the prior convergence rate, Genetic Algorithm evolution process adopts from the pseudo-biological operations namely selection, crossover, mutation, and other extra operations. The determination of the exact local optimal solution is poor in GA whenever the algorithm integrates with another algorithm. But, it is a proficient method in discovering the whole investigation cosmos.

Dr. James Kennedy, and Dr. Russell Elberhart, [7] developed Particle Swarm Optimization (PSO) in 1995. PSO is a complementary technique to GA. It is one of the optimum models provides vulnerable communication between the independent agents. It examines the parameter space globally by controlling the paths of the group of elements like a swarm by collecting the social knowledge from the individual [8]. Because of its convergence speed and easiness, it is recognized as a global optimization technique. PSO exhibited well earlier convergence rate than GA. PSO is failed in discovering enhanced solution during reproduction runs.

3. PROPOSED METHODOLOGY

3.1 Various SRGMs used

1. Goel Okumoto Model [9]

$$m(t)=a(1-e^{-bt}) \quad (1)$$

2. Delayed S-shaped Model [9]

$$m(t) = a(1-(1+bt)e^{-bt}) \quad (2)$$

3. Huang Logistic Model [10]

$$L(t) = \sum_{n=1}^{k} \mu_{cf_k} / \mu_{f_k} \quad (3)$$

Where

μ_{cf} - mean value of the cumulative number of failures

μ_{f_k} - Mean value of failure.

4. Yamada Exponential [11]

$$m(t) = a(1-e^{-r\alpha(1-e^{-\beta t})}) \quad (4)$$

5. Inflection S-shaped Model [9]

$$m(t)= \frac{a(1-e^{-bt})}{1+\beta e^{-bt}} \quad (5)$$

6. Yamada Imperfect Debugging Model 1 [12]

$$m(t) = \frac{ab(e^{\alpha t}-e^{-bt})}{a+b} \quad (6)$$

7. Yamada Rayleigh Model [11]

$$m(t) = a(1-e^{-r\alpha(1-e^{-\beta t^2/2})}) \quad (7)$$

8. Yamada Imperfect Debugging Model 2 [12]

$$m(t)= a(1-e^{-bt})(1-\frac{\alpha}{b})+\alpha at \quad (8)$$

9. Two-Dimensional S-shaped Model [10]

The Mean value function is detected is given by,

$$F_N = \mu_f - \mu_{fr} \quad (9)$$

Where

μ_f -Mean number of faults detected with respect to the Coverage and time

μ_{fr} -Mean number of failures with respect to the Coverage and time

10. P-N-Z Model [13]

$$m(t)= \frac{a(1-e^{-bt})(1-\frac{\alpha}{b})+\alpha at}{1+\beta e^{-bt}} \quad (10)$$

3.2 Comparison criteria

In order to evaluate the performance of SRGMs need to estimate the various parameters which help in reliability estimation. Some of the parameters are given below.

1. Cumulative Number of Failures

$$C(f)= (a(1-r))^M/1-r \quad (11)$$

where,

a is failure number

r is time

M is max number failure time

2. Mean value of number of faults [14]

$$m(f) = R(1 - \exp(-n * T))/(1 + \beta * \exp(-n * T) \quad (12)$$

3. Software reliability measure [14]

$$R_s(\varphi / T_\pi, C_\pi) = \exp\{-[M_f((T_\pi + \varphi), C_\pi / h) - M_f(T_\pi, C_\pi / h)]\} \quad (13)$$

3.3 SRGM selection using MGSO

The proposed Modified Genetic Swarm Optimization (MGSO) method is a hybrid Optimization technique, which abuses the novelty and individualism of two Orthodox optimization gradients in the most efficient way. They are Particle Swarm Optimization (PSO) and Genetic Algorithm (GA).

3.3.1 Modified Genetic Swarm Optimization (MGSO)

In order to solve the complex combinatorial optimization problems, that involves the problems of convergence speed and consistency in the solution space. There is a necessity to develop a potent scheme that overcomes the problems of consistency and convergence speed. Genetic Swarm Optimization (GSO) is a population-based pragmatic search method, proved on the notions of biological selection and evolution, depending on traditional and social guidelines resulted from the study of the swarm intelligence and from the interface among particles [15]-[16]. Both PSO and GA show variances in an estimation of performance measures such as consistency and convergence speed. A different search method is defined by combining some of the features such as selection-crossover-mutation from GA and velocity update from PSO; both the search methods show a gallant performance for some applications. But, both the algorithms showed similar results because of their natural population-based intent of parameters. Hence, it is concerned to develop a hybrid method to use the competencies and nuttiness of both the algorithms. GSO involves a vigorous support of GA and PSO since it endorses the incorporation of both the procedures for the complete execution. Sometimes, both the methods show similar behavior in the selection of the fitness or for better information sharing.

The pictorial representation of integrating PSO with GA algorithms is shown in below fig. 1.

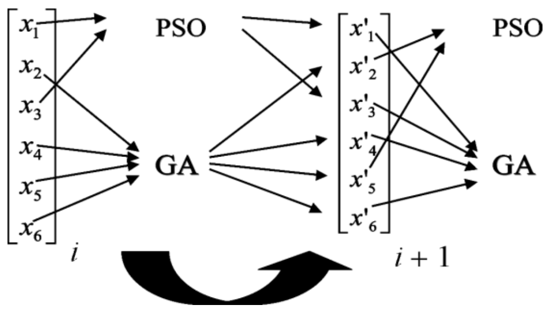

Fig 1: Merging of GA and PSO

The idea of MGSO is instigating with one technique and then applying another technique on the obtained results from the first technique. The operation of one technique is embedding into another technique for example, mutation and crossover operators of Genetic Algorithm applied into Particle Swarm Optimization. Apply the local search on the obtained results from the global search. Test the entire population by dividing into various subpopulations and use algorithms to get more interesting results. In this paper, we focus on hybridization of PSO method with other search methods that are local and global as well. In the proposed scheme we considered Particle Swarm Optimization is the main algorithm, and then merged with other technique Genetic Algorithm.

To optimize the results of the software reliability growth model, the assessed parameters of SRGM coupled with the Genetic Swarm Optimization algorithm, which enhances the estimated parameters like failure recognition rate, fault exclusion, and reliability. The following expression represents the equation for reliability enhancement for a system x.

$$MGSO(x) = GSO(x) + LTEF(x) \quad (14)$$

In each iteration, the given population is separated into two subpopulations and evolved with two algorithms namely Genetic Algorithm and Particle Swarm Optimization. In the newly generated population, the obtained results are then reunified and again divided subjectively into two fragments in the following iteration for the next run of genetic or particle swarm algorithms. In a way, the population appraisal perception is simply known by assuming that a portion of the individuals is replaced by a newly created one using GA, whereas the remaining are of the preceding generation but then stimulated on the solution space using PSO. The core parameter of GSO algorithm is Hybridization Coefficient.

The general steps involved in MGSO are as shown in below fig. 2.

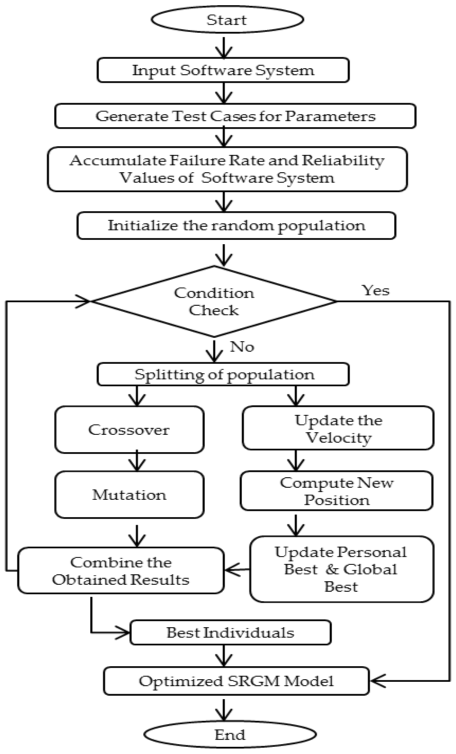
Start
Input Software System
Generate Test Cases for Parameters
Accumulate Failure Rate and Reliability Values of Software System
Initialize the random population
Condition Check
Yes
No
Splitting of population
Crossover
Update the Velocity
Mutation
Compute New Position
Combine the Obtained Results
Update Personal Best & Global Best
Best Individuals
Optimized SRGM Model
End

Fig 2: Flow diagram of MGSO

4. RESULTS AND DISCUSSIONS

Software Reliability Growth Model is a mathematical statistical expression used in software reliability prediction [17]. In this paper, we employed a system, which helps in selection of SRGM using Modified Genetic Swarm Optimization technique. The proposed scheme offers to select an appropriate software reliability growth model which can be used for testing the reliability of any kind of software application. The selection of the model can be performed by estimating reliability parameters and optimizing the obtained

results. In this work, we have exploited two projects for the selection of model. In the following sections, the comparative results of various SRGMs using optimization techniques are explained for the two projects. The proposed methodology implemented using the tools NetBeans and Java platform. The acquired results are assessed using MGSO, PSO, and Genetic Algorithms to study the performance of the proposed methodology.

The values of the various parameters for the ten NHPP SRGMs have been evaluated using the maximum likelihood estimation with the incorporation of Logistic testing effort function (LTEF). Because of insufficient space we are providing some of the parameters which are playing vital role in reliability prediction are the number of failures decreased and reliability are estimated at different time intervals are provided in Table 1 and Table 2.

The following table 1 shows the number of failures occurred at different time intervals during the testing phase and the optimized results using MGSO, GA, and PSO optimization techniques.

Table 1. Decreased failure rate for various optimization techniques at different time instants

Time (S)	# Failures Decreased			
	Before Optimization	After GA	After PSO	After MGSO
1	16	13	14	11
5	25	18	17	17
10	48	36	37	34
20	21	16	16	15
25	32	23	27	22

Fig 3 shows the comparative results of the number of failures decreased using MGSO, GA, and PSO optimization techniques at different time intervals.

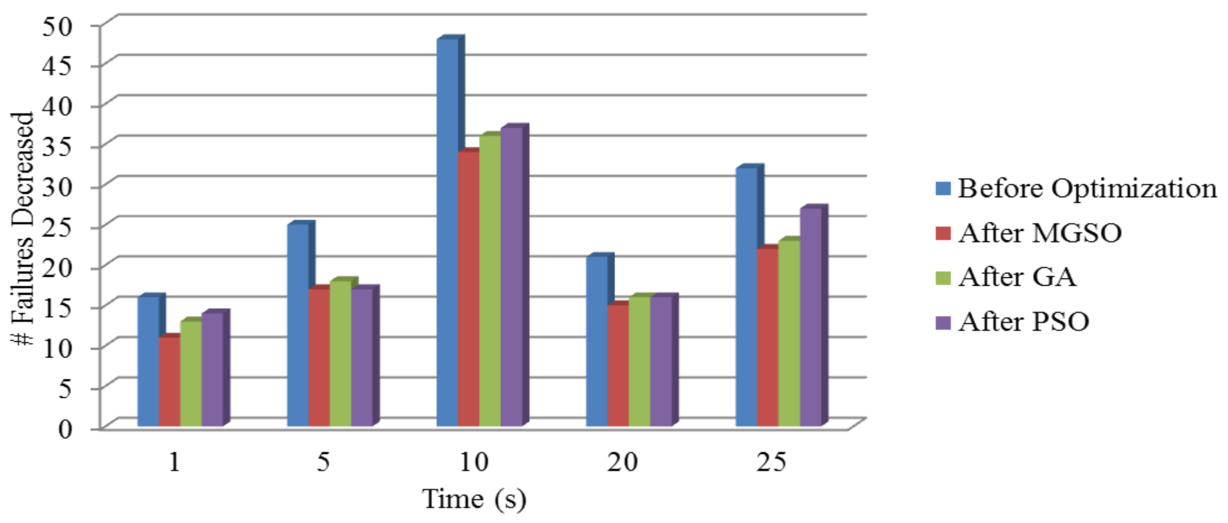

Fig 3: Graphical representation of # failures decreased at different time intervals using various optimization techniques

The following Table 2 shows the reliability calculations at different time intervals during the testing phase and the optimized results using MGSO, GA, and PSO optimization techniques.

Table 2. Comparison of reliability parameter using various optimization techniques at different time instants

Time (S)	Reliability			
	Before Optimization	After GA	After PSO	After MGSO
1	0.136	1.118	0.295	1.038
5	0.248	0.37	0.332	1.894
10	0.125	1.153	0.135	1.868
20	0.108	0.745	0.492	0.897
25	0.187	0.438	0.285	0.63

Fig 4 shows the comparative results of Reliability parameter using MGSO, GA, and PSO optimization techniques at different time intervals.

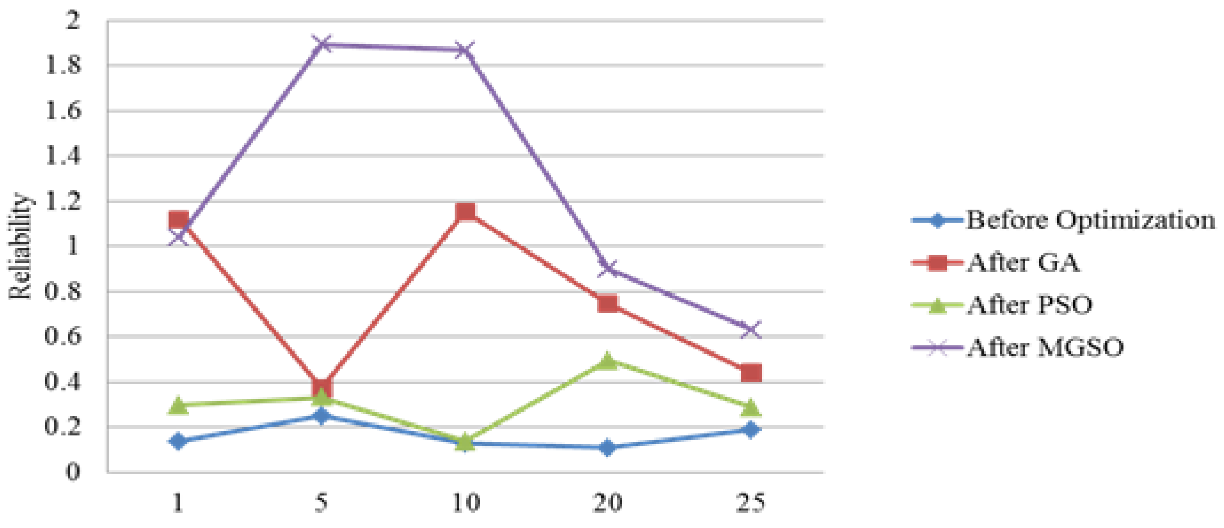

Fig 4: Comparison of Reliability parameter at different time intervals using various optimization techniques

4.1 Application Examples

Example 1: A dataset (DS1) in the experimental evaluation is used hospital health care system application. For the hospital health care system application, the number of failures decreased, cumulative number of failures, mean value function, reliability and MLE fitness are measured. The proposed system employs Modified GSO as the optimization technique and the values that are obtained for different models compared. The Table 3 given below shows the best performance values of different software reliability growth models using three optimization techniques such as Genetic algorithm, Particle swarm optimization, Modified genetic swarm optimization. From the obtained values we can say that Huang logistic model with modified genetic swarm optimization technique yields better results than the other software reliability growth models. Here, the maximum value considered as the best value for the feasible solution.

Example 2: Second dataset (DS2) is used in the experimental evaluation is banking application system. For the banking application system, the number of failures decreased, cumulative number of failures, mean value function, reliability and MLE fitness are measured. The proposed system employs Modified GSO as the optimization technique, and the values that are obtained for different models compared. The Table 4 given below shows the best performance values of different software reliability growth models using three optimization techniques such as Genetic algorithm, Particle swarm optimization, Modified genetic swarm optimization. From the obtained values we can say that Huang logistic model with modified genetic swarm optimization technique yields better results than the other software reliability growth models. Here, the maximum value considered as the best value for the feasible solution.

5. CONCLUSION

In this paper, comparison of various SRGMs and selection of software reliability growth model (SRGM) in which Modified Genetic Swarm Optimization technique is proposed. The various parameters of different software reliability growth models have been employed using optimization techniques. Also, the proposed optimization technique helps in selection of SRGM by calculating the efficiency (fitness) function. Once the efficiency is calculated and it is enhanced using the optimization technique. The highest efficiency value is the best SRGM. Here we have used Modified GSO where genetic operators of one technique are incorporated in another technique in order to deliver the better selection of models. The implementation results have revealed the effectiveness of the proposed method of selecting appropriate software reliability growth model. The efficiency of the model concerning decreased failure rate and optimizing the fitness are the major consideration for selecting the appropriate model for reliability growth in our proposed method.

6. REFERENCES

[1] M.R.Lyu, "Handbook of Software Reliability Engineering," Mc Graw Hill, 1996

[2] Kapil Sharma, Rakesh Garg, C.K. Nagpal, and R.K.Garg, "Selection of Optimal Software Reliability Growth Models Using a Distance Based Approach," IEEE Transactions on Reliability, Vol. 59, No.2, pp 266-276, June 2010.

[3] Chin-Yu Huang, and Michael R. Lyu, "Estimation and Analysis of Some Generalized Multiple Change-Point Software Reliability Models," IEEE Transactions on Reliability, Vol. 60, No.2, pp 498- 514, June 2011.

[4] P.K. Kapur, H.Pham, Sameer Anand, and Kalpana 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, March 2011.

[5] Y.P. Wu, Q,P.Hu, M.Xie, and S.H. Ng, " Modeling and Analysis of Software Fault Detection and Correction Process by Considering Time Dependency," IEEE Transactions on Reliability, Vol. 56, No. 4, pp. 629-642, December 2007.

[6] Ali Hadidi, Sina Kazemzadeh Azad, Saeid Kazemzadeh Azad, "Structural optimization using artificial bee colony algorithm", *Proc.of International Conference on Engineering Optimization*, 2010.

[7] J. Kennedy and R. C. Eberhart, "Particle swarm optimization", *Proc. of IEEE International Conference on Neural Networks*, Vol. 5, No. 3,pp. 1942–1948, 1995.

[8] Ali R. Yildiz, "Cuckoo search algorithm for the selection of optimal machining parameters in milling operations", *The International Journal of Advanced Manufacturing Technology*, Vol.64, Issue.1-4, pp.55-61, 2012.

[9] C. Y. Huang, M.R. Lyu, and S.Y. Kuo, " A unified scheme of some non-homogeneous Poisson process models for software reliability estimation," IEEE Transactions on Software Engineering, Vol. 29, No.3, pp. 261-269, March 2003.

[10] Mallikharjuna Rao. K, Anuradha. Kodali, "An Efficient Method for Software Reliability Growth Model Sselection using Modified Particle Swam Optimization Technique", International Review on Computers and Software (I.RE.CO.S), Vol. 10, No.12, pp. 1169-1178, December 2015.

[11] H.Pham, "Software reliability and cost models: perspectives, comparison and practice," European Journal of Operational Research, Vol. 149, pp. 475-489, 2003.

[12] S. Yamada, K.Tokuno, and S.Osaki, "Imperfect debugging models with fault introduction rate for software reliability assessment," International Journal of System Science, Vol. 23, No. 12, 1992.

[13] H.Pham, L. Nordmann, and X. Zhang, "A general imperfect software debugging model with s-shaped fault detection rate," IEEE Transactions on Reliability, Vol. 48, pp. 169-175, June 1999.

[14] Mallikharjuna Rao. K, Kodali Anuradha, "An Efficient Method for Parameter Estimation of Software Reliability Growth Model Using Artificial Bee Colony Optimization," In Proc. Of 5th International Conference on SEMCCO-2014, Lecture Notes in Computer Science, Springer Series, Vol. 8947, pp. 765-776, Switzerland 2015.

[15] Anupriya, Akashtayal, "Comparison of Hybrid and classical Metaheuristic for Automatic image enhancement", *International Journal of Computer Applications*, Vol. 46, No.2, pp. 0975 – 8887, May 2012.

[16] Apurba Gorai, Ashish Ghosh, "Hue-Preserving Color Image Enhancement Using Particle Swarm Optimization", *Proc. 0f Recent Advances in Intelligent Computational Systems*, pp. 563 - 568, Sept 2011.

[17] S. Yamada, H. Ohtera and R. Narihisa, "Software Reliability Growth Models with Testing-Effort," IEEE Trans. Reliability, Vol. R-35, pp.19-23 (1986).

7. APPENDIX

Table 3: Comparison of various SRGMs using Various Optimization Techniques

Cumulative #		Mean Value Function m(t)				Reliability R(t)				Fitness			
After PSO	After MGSO	Before Optimization	After GA	After PSO	After MGSO	Before Optimization	After GA	After PSO	After MGSO	Before Optimization	After GA	After PSO	After MGSO
1.151	0.318	2.54	1.70	0.97	0.159	1.627	1.76	0.82	0.34	15442	783.253	1226.419	593.659
1.16	0.47	2.18	1.53	0.92	0.13	0.617	0.31	0.3119	0.3092	15442	1574.542	1341.658	1299.876
0.87	0.15	2.49	1.80	0.841	0.456	1.427	0.72	1.122	0.106	112701	230526.453	300537.238	300537.238
0.884	0.48	1.98	1.29	1.25	0.44	1.0311	1.24	0.67	0.14	15442	1341.658	1268.856	1139.452
0.75	0.17	2.02	1.517	1.528	0.4136	1.288	1.19	1.08	0.189	15442	9785.267	205293.1417	205293.1417
1.393	0.461	17.52	1.38	1.53	0.2811	1.853	1.23	0.848	0.4307	15442	1759.349	1924.323	1676.7567
1.03	0.417	9.22	1.36	1.35	0.199	2.327	1.24	1.36	0.294	15442	1341.658	1924.323	968.4864
1.118	0.232	8.662	1.77	0.78	0.16	1.160	1.28	1.359	0.202	15442	1341.658	1815.57	1209.7576
1.207	0.403	4.652	0.74	1.12	0.366	2.092	1.48	0.983	0.232	15442	5.127 xE^7	4.76x E^9	4.76x E^9
1.47	0.47	2.52	1.38	1.34	0.43	1.222	1.72	0.644	0.102	15442	1924.323	1226.419	1189.7536

Table 4 : Comparison of various SRGMs using Various Optimization Techniques

Cumulative # of			Mean Value Function m(t)				Reliability R(t)				MLE Fitness				Model	# of Failures Decreased					
After GA	**After PSO**	**After MGSO**	**Before Optimization**	**After GA**	**After PSO**	**After MGSO**	**Before Optimization**	**After GA**	**After PSO**	**After MGSO**	**Before Optimization**	**After GA**	**After PSO**	**After MGSO**		**Before Optimization**	**After GA**	**After PSO**	**After MGSO**	**Before Optimization**	**After GA**
1.02	1.91	0.13	9.37	1.56	1.001	0.31	1.48	0.73	0.95	0.109	142	132.57	128.8	119.17	Goel Okumoto	48	35	34	34	2.096	1.49
1.73	1.308	0.39	2.04	1.21	0.87	0.46	2.71	1.33	0.79	0.308	142	139.479	116.45	106.069	Delayed S-shaped	48	34	36	34	2.08	1.52
1.38	0.98	0.33	8.27	1.50	1.51	1.35	2.11	1.35	1.34	0.256	1794.753	7206.251	5785.224	6893.347	Huang Logistic	48	38	34	32	2.36	1.27
1.81	1.29	0.44	9.13	1.94	1.79	1.12	1.71	1.17	0.99	0.332	142	118.975	128.88	111.532	Yamada Exponential	48	36	34	34	3.20	1.99
1.20	0.67	0.32	5.54	1.13	0.26	0.14	1.16	0.71	0.39	0.133	1892.4795	4527.872	5990.144	3557.177	Inflection S-shaped Model	48	39	36	31	1.89	0.74
1.59	1.59	0.76	9.54	1.56	0.80	0.28	1.68	1.17	0.87	0.446	150	133.658	123.88	117.152	Yamada Imperfect Debugging 1	48	36	35	34	2.68	0.597
1.79	1.32	0.46	7.08	1.73	1.58	0.44	1.84	1.75	1.08	0.35	142	123.753	141.508	110.742	Yamada Rayleigh	48	34	35	34	2.91	1.57
1.58	1.19	0.47	7.58	1.80	1.62	0.48	1.74	0.78	1.17	0.18	142	125.35	134.637	103.719	Yamada Imperfect Debugging 2	48	35	37	34	2.46	1.6
1.32	0.73	0.34	5.60	1.35	1.18	0.6	1.34	0.38	0.4	0.22	142	4032.799	4032.799	3889.747	Two-Dimensional S-shaped Model	48	38	36	33	3.23	2.004
1.306	1.79	0.47	9.37	2.17	1.97	0.45	1.15	0.68	0.32	0.105	150	136.708	123.19	118.176	P-N-Z Model	48	37	36	34	3.054	1.77

Model	# of Failures Decreased				
	Before Optimization	After GA	After PSO	After MGSO	Before Optimization
Goel Okumoto	25	21	21	17	7.24
Delayed S-shaped	48	35	38	34	4.44
Huang Logistic	32	25	25	22	7.78
Yamada Exponential	25	20	20	17	7.78
Inflection S-shaped Model	32	25	25	22	7.78
Yamada Imperfect Debugging 1	48	36	36	34	4.65
Yamada Rayleigh	21	18	16	15	5.79
Yamada Imperfect Debugging 2	32	24	26	22	4.65
Two-Dimensional S-shaped Model	32	25	25	22	7.78
P-N-Z Model	48	36	35	33	5.704