Applications and Applied Mathematics: An International Journal (AAM)

Volume 19
Issue 3 *Special Issue No. 12 (March 2024)*

Article 5

3-2024

Fuzzy Software Reliability and Optimal Release Policy with Log-Logistic Testing Effort: An Analysis

Seema Rani
T. M. Bhagalpur University, Bhagalpur

Jitendra Kumar
T. M. Bhagalpur University, Bhagalpur

N. Ahmad
T. M. Bhagalpur University, Bhagalpur

Available at
http://pvamu.edu/aam
Appl. Appl. Math.
ISSN: 1932-9466

Applications and Applied Mathematics:
An International Journal
(AAM)

Special Issue No. 12 (March 2024), Article 5, 12 pages

Fuzzy Software Reliability and Optimal Release Policy With Log-Logistic Testing Effort: An Analysis

[1*]Seema Rani, [1,2]Jitendra Kumar, and [1,3]N. Ahmad

[1]University Department of Statistics and Computer Applications
T. M. Bhagalpur University, Bhagalpur
Bihar, India

[1*]seema17.rani@gmail.com; [2]jitendra12.jgd@gmail.com; [3]ahmad_n@tmbuniv.ac.in

[*]Corresponding Author

Received: May 1, 2023; Accepted: September 18, 2023

Abstract

We will discuss a Software Reliability Growth Model (SRGM) using fuzzy and imperfect debugging environments; we integrate Log-Logistic (LL) Testing Effort Function (TEF) into fuzzy SRGMs. Estimation methods, such as Least Square and Maximum Likelihood, are used to obtain the value of Testing-Effort and SRGMs parameters. It is not always possible and is constantly required to quantify the exact value of parameters. Due to human conduct, the value of Testing-Effort and SRGM parameters cannot be exactly quantified. In this scenario, parameters are supposed to be vague or fuzzy. To make the software consistent, the developer needs to propose some quantity of vagueness. Therefore, we propose a reliability growth model together with suspicions involved in parameters of SRGM under imperfect debugging using fuzzy theory. We calculate the entire cost of software development and reliability of proposed model using Triangular Fuzzy Number (TFN) for real data sets. Results obtained are compared with previous works from literature. It is shown the proposed SRGM with LL TEF give realistic extrapolation ability of software reliability and optimized entire software cost under fuzzy environment.

Keywords: Software Reliability Growth Model; LL Testing Effort Function; Fuzzy number; Fuzzification; Defuzzification; Triangular Fuzzy Number (TFN); Fuzzy set theory

MSC 2010 No.: 68M15, 68N30, 94D05

1. Introduction

We know that human nature is uncertain. Interaction between humans and the real world raise uncertainty. Whenever a person does not qualitatively hold the appropriate information to explicate its presentation or additional phenomena deterministically and numerically, then uncertainty appears. The modern vision of uncertainty was introduced in the late 19^{th} century, and it started from conversion from the conventional vision. Replacement of Newtonian mechanism, which includes only certainty from probability theories, is known as first stage of conversion. Zadeh (1965) produced a paper on fuzzy set which started the modern view of uncertainty, known as second stage of conversion. Probability theory captured uncertainty of confident type. This concept of uncertainty is challenged by Zadeh (1965). Idealized uncertainty arises everywhere. It is elementary scientific standard, including necessary and ubiquitous phenomenon (Gong and Hai (2014)).

A customer always requires software with no defects and minimum cost at specific time. Estimation of software reliability is a very significant task in software development. Software Reliability Growth Models (SRGM) is a mathematical tool used to guess and estimate the reliability of software (see Yamada et al. (1984), Musa et al. (1987), Lyu (1996)). Using dissimilar postulation and surroundings, varieties of SRGM are developed in past decades (see Musa (1975), Goal and Okumoto (1979), Obha (84), Yamada and Osaki (1985), Yamada et al. (1993), Lyu (1996), Ahmad et al. (2008)). These SRGM provide essential information for making decision in the processes of software development, such as testing resource/effort consumption (see Yamada et al. (1993), Huang (2005), Bokhari and Ahmad (2006), Ahmad et al. (2011)) and imperfect debugging (see Rafi et al. (2010), Ahmad et al. (2021)). Optimal release policy and SRGM with different testing-effort considering imperfect debugging have been introduced by many authors (see Kapur et al. (2007), Ahmad et al. (2008), Bokhari and Ahmad (2014), Rafi et al. (2010), Huang and Lyu (2005)). Change point perspective has also been considered into SRGM (see Huang (2005)). Soft computing techniques have been used to estimate the parameters of SRGM by several researchers (see Sheta (2007), Kiran and Ravi (2007), Jin and Jin (2016), Shailee et al. (2019)).

Further, development of software is explained by many parameters. Every parameter included definite level of fuzziness. So, it becomes necessary to consider the degree of uncertainty involved in these parameters to develop more reliable software. During the operational and testing phase of the software development, a tool is used for measuring software reliability, called SRGM (Kapur et al. (2007)). Also, uncertainty exists in reliability estimation, expenditure estimation, effort evaluation and risk study in software expansion procedure. In this scenario, several works have been done under fuzzy environment for last two decades (see Klirr and Yuan (1995), Kapur et al. (2011), Jha et al. (2011), Pachauri et al. (2013), Rani et al. (2016), Kalayathankal et al. (2017), Dwivedi and Kumar (2018), Kumar and Ram (2018), Rani and Ahmad (2019, 2020a), Lee et al. (2022), Kumar et al. (2021)). Recently, Rani and Ahmad (2020) also discussed a SRGM considering testing-effort functions under fuzzy theory to predict the failure of the software.

This paper integrated Log-Logistic Testing-Effort (LLTE) into fuzzy SRGM. Two methods are used to estimate the LLTE and SRGM parameters, named as least square estimation and maximum likelihood methods. Data analysis is present numerically. This paper is an extension of Bokhari and Ahmad (2006), Ahmad et al. (2011), Ahmad et al. (2008) and Ahmad et al. (2021) to the fuzzy environment. This work is little bit different from the work of Rafi et al. (2010) and Rani and Ahmad (2019).

Proposed work is divided into five sections. LLTE and the proposed model is described in Section 2, and parameter assessment, fuzzification of estimated parameters, arithmetic operation, defuzzification of fuzzy reliability and comparison between obtained result and the previous model are described in Section 3. Optimal Release Policy in fuzzy environment is described in Section 4. This work is concluded in Section 5.

2. Fuzzy Software Reliability Growth Model

A. Log-Logistic Testing-Effort

During the testing phase of software development, lots of TE is consumed. The expanded TE helps us to identify the mistakes successfully and can be described by different circulation (Kapur et al. (2007), Yamada et al. (1993)). SRGM included LLTEF can determine maximum number of faults compare than previous methodologies. We are producing fuzzy SRGM using LLTEF.

We can define the cumulative testing-effort of LLTEF in time (0,t] as Ahmad et al. (2008):

$$W(t) = \eta\left[1 - \frac{1}{1+(\text{Ɑ}t)^{\mu}}\right] = \eta\left[\frac{(\text{Ɑ}.t)^{\mu}}{1+(\text{Ɑ}.t)^{\mu}}\right], \ t > 0, \tag{1}$$

where η is total TE consumed by software testing Ɑ and μ are scale and shape parameters, respectively.

And current testing-effort at time *t* can be defined as:

$$w(t) = W'^{(t)} = \frac{\eta.\text{Ɑ}.\mu(\text{Ɑ}t)^{\mu-1}}{[1+(\text{Ɑ}t)^{\mu}]^2}, \ t > 0, \eta > 0, \text{Ɑ} > 0, \mu > 0. \tag{2}$$

Maximum testing-effort $w(t)$ at *t* is:

$$t_{max} = \frac{1}{\text{Ɑ}}\left(\frac{\eta-1}{\eta+1}\right)^{\frac{1}{\eta}}. \tag{3}$$

B. Proposed Fuzzy SRGM Model

The following are the assumptions of proposed SRGM based on the model described in Rafi et al. (2010):

1. The NHPP concept is used to detect and eliminate the error in testing phase of software development.
2. It is obvious that software failure occurs at random time.
3. Whenever a failure arises, the error due to which it is directly detached and novel errors may be familiarized in among the fault decrease process with some prospect, say ɓ.
4. The time interlude (t, $t + \Delta$t) to the obtainable TE is trained to the mean number of enduring errors in the system and proportionality is perpetual finished time in the number of faults identified.
5. Log-Logistic TE function is implemented.

As above assumption mathematical representation of proposed model is:

$$\frac{dm(t)}{dt} . \frac{1}{w(t)} = n\big(a(t) - m(t)\big), \tag{4}$$

$$\frac{da(t)}{dt} = ɓ \frac{dm(t)}{dt}, \tag{5}$$

where $m(t)$ denotes mean value function, $w(t)$ is Testing-Effort Function (TEF), $a(t)$ is the total number of errors, n is the error detection rate, and ɓ is the probability of introducing new error.

At $m(0) = 0, W(0) = 0$, and $a(0) = a$, mean value function $m(t)$ can be represented as:

$$m(t) = \frac{a}{1-ɓ}\left(1 - e^{-n.(1-ɓ).W(t)}\right). \tag{6}$$

Finally, the reliability function can be defined as:

$$R(t + \Delta t/t) = e^{-m(t+\Delta t) - m(t)}. \tag{7}$$

3. Data Investigation

A. Parameter Assessment

Parameters are the total amount of testing-effort expenditure (η) required by software testing, scale parameters (Ƨ), shape parameters (μ), total number of faults (a), detection rate (n), and probability to introduce new faults (ɓ) of SRGM with Log-logistic testing effort can be expected by using actual data sets and applying the following two estimation methods.

B. Least Square Estimation (LSE)

Let *n* experimental data couples be in the form (p_i, Q_i) where $\left(i = 1, 2, \dots\dots q; 0 < p_1 < p_2 < \dots < p_q\right)$ where Q_i is the cumulative TE consumed in time $(0, p_i)$. Then, the parameters of LL testing-effort are predicted by minimizing (Ahmad et al. (2008)):

$$S(\eta, \text{Ƣ}, \mu) = \sum_{i=1}^{p} [Q_i - Q(p_i)]^2.$$

C. Maximum Likelihood Estimation

The value of parameters $\hat{\eta}$,$\hat{\text{Ƣ}}$, and $\hat{\mu}$ have been obtained by the method of least square discussed earlier. Let q detected data couples are in the form (p_i, x_i) where $\left(i = 1, 2, \dots\dots q; 0 < p_1 < p_2 < \dots < p_q\right)$ and x_i is detected cumulative number of faults during $(0, p]$. This method predicated the parameters a, n and ɓ in the SRGM model by MLE method (Ahmad et al. (2008)):

$$L(a, n, \text{ɓ}) \equiv P\{N(p_i) = x_i, i = 1, 2, \dots\dots\dots, q\},$$

$$= \prod_{i=1}^{q} \frac{[m(p_i) - m(p_{i-1})]^{(x_i - x_{i-1})}}{(x_i - x_{i-1})!} . e^{-[m(p_i) - m(p_{i-1})]},$$

where $p_0 \equiv 0$ and $x_0 \equiv 0$.

The data of Tohma et al. (1989) have been used in our proposed model. This data set is taken after 22 days of testing, a whole of 86 software faults were identified and 93 CPU hours were expended. We estimated the SRGM parameters through MLE method.

To approximate the factors η, Ƣ, and μ of Log-Logistic TE, we fit the real TE data into Equation (1) and (2) and resolve it by the LSE method. These expected parameters are:

$$\eta = 177.02, \; \text{Ƣ} = 0.048, \; \mu = 1.973.$$

By using these estimated TE parameters η, Ƌ, and μ, the SRGM parameters *a, n,* and ɓ in Equation (6) can be solved by MLE method. We get following estimation for SRGM parameters through SPSS; these are:

$$a = 133.1, \; n = 0.016, \quad ɓ = 0.265.$$

D. Fuzzification of Estimated Parameters

The process of converting crisp erratic into fuzzy variable is called Fuzzification. Intuition, experience, and analysis of the set of rules and conditions related with the input variables are generally used fuzzification method. Here, intuition method of fuzzification is used to fuzzified the crisp erratic into fuzzy variable.

The process of fuzzification can be illustrated by this example as follows below.

Illustrated Example: Assume the crisp value of parameter '*a*' is 133.1. For example, *a*= 133.1 and *w*e get 1% of *a* equal to 1.331. Now, the resultant value added and subtracted with the original value of a to get the value of $a_4 = 133.1 + 1.331 = 134.431$ and $a_2 = 133.1 - 1.331 = 131.769$ respectively. To find $a = 130.438$ again, we subtract the value of 1% of 133.1 from a_2=131.769. The above illustration can be summarized as follows:

$$\left\{133.1 - \left(1 * \left(\frac{133.1}{100}\right)\right\} - \left(1 * \left(\frac{133.1}{100}\right)\right) = a_1,\right.$$

$$133.1 - \left(1 * \left(\frac{133.1}{100}\right)\right) = a_2,$$

$$133.1 = a_3,$$

$$133.1 + \left(1 * \frac{133.1}{100}\right) = a_4.$$

Triangular fuzzy number (TFN) *z* can be distinct as follows, based on above mentioned procedure:

$$\mu_A(z) = \begin{cases} 0, & x < 131.769, \\ \dfrac{x - 131.769}{133.1 \; - 131.729}, & 131.769 \le x \le 133.1, \\ \dfrac{134.431 - x}{134.431 - 133.1}, & 133.1 \le x \le 134.431, \\ 0, & x > 134.431. \end{cases}$$

By following above procedure, the rest parameters of TE and SRGM have been fuzzified at +/-1% spreads. We fuzzified all parameters, i.e., the total amount of testing-effort expenditure (η) required by software testing, scale parameters (ꞛ), shape parameters (μ), total number of faults (a), detection rate (n) and new fault rate (Ϭ) at +/-2 and +/-3 percent spreads same as +/-1% spreads.

E. Arithmetic Operation

$$q = [b^{\alpha}, d^{\alpha}] = [b + (c - b)\alpha, d - (d - c)\alpha], \quad \forall \alpha \in [0,1]. \tag{8}$$

Triangular fuzzy number max and min bounds of all fuzzified parameters are calculated by using above mathematical expression as Equation (8) at different assumption Level (AL) (0 to 1) (Klirr and Yuan (1995)). Max and min bound of $W(t)$ is calculated from Equation (1) at different AL at $t = 10, 15, 20, 25$ and $t = 10.1, 15.1, 20.1, 25.1$. Here we used Δt = 0.100. Now, we calculated max and min bound of $m(t)$ and $m(t + \Delta t)$ from Equation (6) and reliability from Equation (8). Max and min bound of $m(t)$ and $m(t + \Delta t)$ at $t = 10$ and $t = 10.1$ at different AL (0 to 1) and +/-1, 2 and 3 percent spreads are shown in Table 1 and 2, respectively. The max and min bound of reliability at $t = 10, 15, 20, 25$ for different AL at +/-1, 2 and 3 percent spreads is calculated.

Table 1. Min and Max bound of Mean Value Function at $t = 10$

AL	At +/-1 percent spread		At +/-2 percent spread		At +/-3 percent spread	
	Min bound	*Max bound*	*Min bound*	*Max bound*	*Min bound*	*Max bound*
0.000	3.39	3.56	3.31	3.65	3.23	3.74
0.100	3.39	3.55	3.32	3.63	3.25	3.71
0.200	3.40	3.54	3.34	3.61	3.27	3.68
0.300	3.41	3.53	3.35	3.59	3.30	3.66
0.400	3.42	3.52	3.37	3.57	3.32	3.63
0.500	3.43	3.51	3.39	3.56	3.34	3.60
0.600	3.44	3.50	3.40	3.54	3.37	3.57
0.700	3.44	3.49	3.42	3.52	3.39	3.55
0.800	3.45	3.49	3.44	3.50	3.42	3.52
0.900	3.46	3.48	3.45	3.49	3.44	3.49
1.00	3.47	3.47	3.47	3.47	3.47	3.47

Table 2. Min and Max bound of Mean Value Function at $t = 10.100$

AL	At +/-1 percent spread		At +/-2 percent spread		At +/-3 percent spread	
	Min bound	*Max bound*	*Min bound*	*Max bound*	*Min bound*	*Max bound*
0.000	6.93	7.57	6.63	7.91	6.35	8.27
0.100	6.96	7.53	6.69	7.84	6.43	8.16
0.200	6.99	7.50	6.75	7.77	6.52	8.05
0.300	7.02	7.47	6.81	7.70	6.60	7.94
0.400	7.05	7.43	6.87	7.63	6.69	7.84
0.500	7.08	7.40	6.93	7.57	6.78	7.73
0.600	7.11	7.37	6.99	7.50	6.87	7.63
0.700	7.14	7.34	7.05	7.43	6.96	7.53
0.800	7.18	7.30	7.11	7.37	7.05	7.43
0.900	7.21	7.27	7.18	7.30	7.14	7.34
1.00	7.24	7.24	7.24	7.24	7.24	7.24

F. Defuzzification of Fuzzy Reliability

Transformation of fuzzy variable into crisp variable is called Defuzzification. Centre of Area method, Centre of Sums, Centre of Maxima, and Weighted Average Method are popular defuzzification method. Center of Gravity (CoG) is used for defuzzification and is represented.

Table 3. Comparison of Crisp Reliability with de-fuzzified reliability at different testing time

	Crisp Reliability and Defuzzified Reliability					
		Defuzzified value of Reliability			Pachauri et al. (2013)	Ahmad et al. (2011)
t	*crisp Reliability*	*+/-1 percent*	*+/-2 percent*	*+/-3 percent*		
10	0.46321	0.4685	0.4691	0.4772	0.6565	0.5652
15	0.5801	0.5804	0.5815	0.5826	0.7223	0.7488
20	0.7116	0.7142	0.7232	0.7302	0.7777	0.9129
25	0.8103	0.8109	0.8135	0.8176	0.8234	0.9849

4. Optimal Release Policy

It is more valuable for customers to achieve the software at appropriate time and reasonable cost. Optimal release policy is very important. It helps the developer to release software into market at appropriate time, because it is very important for customer. The developer always prefers to deliver the product at specified time. By using the concept of cost-reliability criterion and fuzzy numbers, Cost model and release policy are discussed in this section. Optimal release policy is also calculated by same process. Bokhari and Ahmad (2006), Ahmad et al. (2011), Pachauri et al. (2013) and Kalayathamkal et al. (2017) redefined the cost function as:

$$Cs(t) = Cs^1 m(t) + Cs^2\{[m(t_{LC}) - m(t)]\} + Cs^3 \int_0^t w(x)dx, \quad (10)$$

where $Cs^1 = (Cs_1^1, Cs_2^1, Cs_3^1)$ the error correction cost during testing, $Cs^2 = (Cs_1^2, Cs_2^2, Cs_3^2)$ is error correction cost during operation, $Cs^2 > Cs^1, Cs^3 = (Cs_1^3, Cs_2^3, Cs_3^3,)$ is the per unit testing-effort expenditures cost and t_{LC} is the length of software life-cycle. In a specified period of time, the conditional reliability of software system is given below:

$$R(t + \Delta t/t) = e^{-m(t+\Delta t) - m(t)}, \quad (11)$$

where $m(t)$ is the mean value function. According to Ahmad et al. (2008), reliability time is calculated to achieve the desired reliability. Total software development cost is calculated at that reliability time. There is need of minimum 24.42781 weeks for testing to get more than 95% reliable software by cost-reliability criterion, as obtained result. The selection of spreads decides the main threat to validate the approach. A wrong selection of spread may guide to incorrect implication. Therefore, it is essential for the system analyst to select appropriate spread while fuzzifying the crisp inputs. The selection of spread may depend on a variety of factors, such as: impreciseness, ambiguity, and accessibility of data, background of operation, good information of the system.

We assume $Cs^1 = 1, Cs^2 = 50, Cs^3 = 100$ results show that, we need minimum 20.6978 weeks for testing to get more than 95% reliable software by cost-reliability criterion. We fuzzify Cs^1, Cs^2 and Cs^3 at plus/minus 1, 2, and 3 % spreads to determine software cost at $t = 24.42781$. Now we calculate min and max bound of Cs^1, Cs^2, and Cs^3. At $t = 24.42781$ we find out the value of $m(T)$ and $W(T)$. We get the value of total cost after putting the min and max bound of Cs^1, Cs^2, Cs^3, $m(T)$ and $W(T)$ in Equation (10). De-fuzzified values of total cost are 15090.6037, 15102.6037, and 15122.1366.

5. Conclusion

In this paper, we have discussed a software reliability modeling and analysis with LLTF using fuzzy set under imperfect debugging environment. We have considered the uncertainty involved in the parameters of SRGM using Triangular Fuzzy Number. Using the proposed method, we obtained the fuzzy reliability and total software cost based on cost-reliability-criterion. We have compared the results with other existing models from the literature. It is revealed that the proposed fuzzy SRGM has a better prediction of reliability measures.

Therefore, we conclude that the modeling of software reliability and software cost may be more effective under the fuzzy paradigm. The proposed fuzzy SRGM may be helpful for software engineers in predicting software reliability measures. Under this framework, it may be extended by incorporation change-point as well as learning factor.

Acknowledgment

The authors are grateful to the editor and two reviewers for their valuable comments and suggestions that lead to improve our manuscript.

REFERENCES

Ahmad, N., Ahmad, A. and Farooq, S. U. (2021). An Assessment of Incorporating Log-Logistic Testing Effort into Imperfect Debugging Delayed S-Shaped Software Reliability Growth Model, International Journal of Software Innovation (IJSI), Vol. 9, No. 3, pp. 23-41.

Ahmad, N., Bokhari, M. U., Quadri, S. M. K. and Khan, M. G. M. (2008). The Exponentiated - Weibull Software Reliability Growth Model with various testing-efforts and optimal release policy: A performance analysis, International Journal of Quality & Reliability Management, Vol. 25, No. 2, pp. 211-235.

Ahmad, N., Khan, M. G. M. and Rafi, L. S. (2011). Analysis of an inflection S-shaped software reliability models considering log-logistic testing-effort and imperfect debugging, International Journal of Computer Science and Network Security, Vol. 11, No. 1, pp. 161-171.

Bokhari, M. U. and Ahmad, N. (2006). Analysis of a Software Reliability Growth Models: The case of log-logistic testing-effort function, In Proceeding of 17th IASTED International Conference on Modeling and Simulation, Montreal, Canada, pp. 540-545.

Bokhari, M. U. and Ahmad, N. (2014). Incorporating Burr Type XII Testing Effort into Software Reliability Growth Modeling and actual Data Analysis with Application, Journal of Software, Vol. 9, No. 6, pp. 1389-1400.

Dwivedi, A. and Kumar, D. (2016). Optimal Release Policy of software with imperfect debugging and testing effort under fuzzy environment, International Journal of Engineering Applied Science and Technology, Vol. 1, No. 8, pp. 103-107.

Goel, A. L. and Okumoto, K. (1979). Time-dependent error-detection rate model for software reliability and other performance measures, IEEE Trans. Reliab. Vol. 28, pp. 206-211.

Gong, Z. and Hai, S. (2014). The Interval-Valued Trapezoidal Approximation of Inter-Value Fuzzy Numbers and Its Application in Fuzzy Risk Analysis, Journal of Applied Mathematics, Vol. 4, pp. 369-379.

Huang, C. Y. (2005). Performance analysis of software reliability growth models with testing-effort and change-point, J. Syst. Softw., Vol. 76, No. 2, pp. 181–194.

Huang, C. Y. and Lyu, M. R. (2005). Optimal release time for software systems considering cost, testing-effort, and test efficiency, IEEE Trans. on Reliability, Vol. 54, No. 4, pp. 583−591.

Jha, P. C., Indumati, Singh, O. and Gupta, D. (2011). Bi-criterion release time problem for a discrete SRGM under fuzzy environment, Vol. 3, No. 6, pp. 680-696.

Jin, C. and Jin, S. W. (2016). Parameter optimization of software reliability growth model with S-shaped testing-effort function using improved swarm intelligent optimization, Applied Soft Computing, Vol. 40, pp. 283-291.

Kapur, P. K., Gupta, A. and Jha P. C. (2007). Reliability Growth Modeling and Optimal Release Policy under Fuzzy Environment of N-version Programming System Incorporating the effect of fault removal efficiency, International Journal of Automation and Computing, Vol. 04, No. 4, pp. 369-379.

Kapur, P. K., Pham, H., Gupta, A. and Jha, P. C. (2011). Optimal Release Policy under Fuzzy Environment, International Journal of System Assurance Engineering and Management, Vol. 2, No. 1, pp. 48-58.

Kiran, N. R. and Ravi, V. (2007). Software reliability prediction by soft computing techniques, Journal of System Software, Vol. 81, No. 4, pp. 576-583.

Klirr, G. J. and Yuan, B. (1995). *Fuzzy Sets and Fuzzy Logic: Theory and Applications*, Prentice-Hall of India Private Limited, New Delhi.

Kalayathankal, S. J., John, T. A. and Kureethara, J. V. (2017). An ordered ideal intuitionistic fuzzy software quality model, International Journal of Mechanical Engineering and Technology, Vol. 8, No. 10, pp. 535-546.

Kumar, A., Bisht, S., Goyal, N. and Ram, M. (2021). Fuzzy Reliability Based on Hesitant and Dual Hesitant Fuzzy Set Evaluation, International Journal of Mathematical, Engineering and Management Sciences, Vol. 6, No. 1, pp. 166-179.

Kumar, A. and Ram, M. (2018). System Reliability Analysis Based on Weibull Distribution and Hesitant Fuzzy Set, International Journal of Mathematical, Engineering and Management Sciences, Vol. 3, No. 4, pp. 513-521.

Lee, D. H., Chang, I. H. and Pham, H. (2022). Software Reliability Growth Model with Dependent Failures and Uncertain Operating Environment, Journals of Applied Sciences, Vol. 12, No. 23, pp. 12383.

Lyu, M. R. (1996). *Handbook of Software Reliability Engineering*, IEEE Computer Society Press, McGraw Hill, New York.

Musa, J. D. (1975). A theory of software reliability and its application, IEEE Transactions on Software Engineering, Vol. 1, pp. 312-327.

Musa, J. D., Lannino, A. and Okumoto, K. (1987). *Software Reliability Measurement*, Prediction and Application, McGraw Hill.

Ohba, M. (1984). Software Reliability Analysis Model, IBM Journal Research Develop., Vol. 28, No. 4, pp. 428-443.

Pachauri, B., Kumar, A. and Dhar, J. (2013). Modeling Optimal release policy under fuzzy paradigm in imperfect debugging environment, Information and Software Technology, Vol. 55, No. 11, pp. 1974-1980.

Rafi, S. M. K., Rao, K. N. and Akhtar, S. (2010). Incorporated Generalized modified Weibull TEF into software reliability growth model and analysis of optimal release policy, Computer and Information Science, Vol. 3, No. 2, pp. 145-162.

Rani, S.and Ahmad, N. (2019). Analysis of fuzzy software reliability growth model and optimal release policy with log-logistic testing effort under imperfect debugging, International Journal of Computer Science and Network Security, Vol. 19, No. 7, pp. 185-195.

Rani, S. and Ahmad, N. (2020). Software Reliability Growth Modeling with Burr Type XII using Fuzzy Logic, 5th International Conference on computing, communication, and security (ICCCS), IEEE, pp. 1-5.

Rani, S. and Ahmad, N. (2020a). An Assessment of Software Reliability Growth Model and Optimal Release Policy with Testing Effort under Fuzzy Environment, Solid State Technology, Vol. 63, No. 6, pp. 5989-6002.

Rani, S., Ara, I. J. and Ahmad, N. (2016). Recent Review and Current Issues in Software Reliability Growth Models under Fuzzy Environment, International Journal of Latest Trends in Engineering and Technology, Vol. 6, No. 4, pp. 550-558.

Sheta, A. (2007). Parameter estimation of software reliability growth models by particle swarm optimization, AIML J., Vol. 7, No. 1, pp. 55–61.

Shailee, L. and Sagar, B. B. (2019). Enhancing Software Reliability prediction based on Hybrid Fuzzy K-Nearest Neighbour with Glowworm Swarm optimization (FLNN-GSO) algorithm, International Journal of Recent Technology and Engineering, Vol. 7, No. 6, pp. 535-546.

Tohma, Y., Jacoby, R., Murata, Y. and Yamamoto, M. (1989). Hyper-Geometric Distribution model to Estimate the Number of Residual Software Fault, Proceeding of COMPSAC-89, IEEE CS Press, Orlando, pp. 610-617.

Yamada, S., Hishitani, J. and Osaki, S. (1993). Software reliability growth model with Weibull testing-effort: A model and application, IEEE Transactions on Reliability, Vol. R-35, pp. 100-105.

Yamada, S., Ohba, M. and Osaki, S. (1984). S-shaped software reliability growth models and their applications, IEEE Transactions on Reliability, Vol. 33, No. 4, pp. 289-292.

Yamada, S., and Osaki, S. (1985), "Software reliability growth modeling: models and applications", IEEE Transaction on Software Engineering, Vol. SE-11, No. 12, pp. 1431-1437.

Zadeh, L. A. (1965). Fuzzy set information and computation, Vol. 8, pp. 338-353.