International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

Hybrid Software Reliability Model for Big Fault Data and Selection of Best Optimizer Using an Estimation Accuracy Function

Ms. Shalini Sharma[1], Dr. Naresh Kumar[2], Dr. Kuldeep Sing Kaswan[3]
[1]School of Computing Science and Engineering
Galgotias University, Greater Noida, India
Email Id: shalini.sharma@rediffmail.com
[2]School of Computing Science and Engineering
Galgotias University, Greater Noida, India
Email Id: Naresh.dhull@gmail.com
[3]School of Computing Science and Engineering
Galgotias University, Greater Noida, India
Email Id: kaswankuldeep@gmail.com

Abstract— Software reliability analysis has come to the forefront of academia as software applications have grown in size and complexity. Traditionally, methods have focused on minimizing coding errors to guarantee analytic tractability. This causes the estimations to be overly optimistic when using these models. However, it is important to take into account non-software factors, such as human error and hardware failure, in addition to software faults to get reliable estimations. In this research, we examine how big data systems' peculiarities and the need for specialized hardware led to the creation of a hybrid model. We used statistical and soft computing approaches to determine values for the model's parameters, and we explored five criteria values in an effort to identify the most useful method of parameter evaluation for big data systems. For this purpose, we conduct a case study analysis of software failure data from four actual projects. In order to do a comparison, we used the precision of the estimation function for the results. Particle swarm optimization was shown to be the most effective optimization method for the hybrid model constructed with the use of large-scale fault data.

Keywords-Software Reliability, Hybrid models, Parameter valuation methods, Soft computing optimization

I. INTRODUCTION

The failure-free operation of software under pre-determined times and conditions is referred to as software reliability. Before a piece of software is released, its reliability is checked, which if not managed properly can result in software failure making reliability a crucial factor in software development. If not managed effectively, software defects can result in software failures making reliability prediction a crucial task. There are many models to handle reliability that are available in the literature [2-15], but when big data analytical systems were utilized these models didn't provide accurate predictions [3]. Software reliability prediction has significantly improved when using a hybrid reliability model rather than a classic one. The performance of models depends heavily on the kind of parameter evaluation technique, making it difficult to choose the optimal strategy among the statistical and soft computing approaches.

Hundreds of software reliability growth models have been proposed by scholars in the past (SRGM). An SRGM is a mathematical model that accurately predicts experimental data and is created by tracking the probability density function or growth curve [16]. The failure pattern of the system is used by SRGM to assess the software's reliability. These failure patterns and data trends are used for reliability estimation. Non-homogeneous Poisson Process (NHPP) models and failure rate models are the two categories under which SRGMs are categorized.

The software is treated as an internal structure with interactions from the external world in NHPP models, known as black-box models. The defect information gathered during testing is used to assess the model's parameters and reliability aspects. The primary need for a software model with a good fit is precise parameter evaluation. The parameter values of dependability models are frequently determined using statistical methods like LSE and MLE. LSE is a minimization strategy that reduces the minimal sum of the squared deviation between the estimated and observed value, which is evaluated by fitting a regression line using data points that satisfy the LSE property. In contrast, MLE determines the parameter value that optimizes the function. MLE is a necessity for the chi-square test, Bayesian approaches, modeling of random effects, and

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

inference with missing data since it is sufficient, consistent, efficient, and parameter invariant.

When the density function is complex, nonlinear, and involves a large number of factors, it is difficult to determine parameter values. In such circumstances, we must numerically determine the parameter value that minimizes LSE or maximizes MLE using optimization techniques such as Newton, quasi-Newton, Gauss-Newton, and Levenberg-Marquardt. Due to the nonlinearity and complexity of the model function, it was difficult to estimate the parameter's value because the model didn't provide a single value for a collection of parameters for several guess values. Additionally, with more parameters, it became too difficult to check all possible permutations of estimate values for an ideal answer. Utilizing soft computing techniques to resolve reliability analysis optimization issues has been popular recently. For parameter evaluation, a variety of bio-inspired techniques such as Particle Swarm Optimization, Cuckoo Search, Grey Wolf, Ant Colony, Artificial Bee Colony, and many more are used alone or in combination with genetic algorithms, neural networks, exponential logistic techniques, and traveling salesman problem.

In this study, we created a hybrid model in which resulting faults were not only because of coding errors but also due to induced error in the software brought on by hardware failure or human error. For nonlinear prediction models, a non-homogeneous reliability model (NHPP) is developed. The suggested methodology combines three NHPP models to assess programming faults and other caused software defects. NHPP models heavily rely on time, fault data, and error rate function to assess dependability and employ mean value function to quantitatively characterize the failure phenomenon. These models perform parameter evaluation through the use of least square estimation (LSE) or maximum likelihood estimation (MLE) methodologies. The primary goal is to assess the value of the mathematical model's unknown parameters, which gives minimum error in the output. The optimal outcome is not always achieved using statistical approaches, despite their being the most popular methodologies. When dealing with complex nonlinear equations with many unknown parameters, soft computing approaches are preferred to statistical methods for parameter estimation.
In this paper, we used GA, SA, and PSO methods for parameter evaluation using MATLAB 2022b. Experiments were performed using four datasets and seven comparison criteria. Comparison is carried out to determine the best optimizer for the developed model using Big Data.

The Paper is organized as follows. Section II discusses the literature survey of the hybrid models used for big data. In Section III literature survey of developed models for big data is discussed. In section IV various techniques applied in this paper to obtain results were briefly discussed. Section V discusses the development of a hybrid model along with the methodology used. Section VI presents the experimental work and obtained results. Finally, Section VII concludes the paper.

II. Literature Survey

Big data processing requires a new class of specialized hardware and software due to its unique properties. Simple software processing errors, data errors, and hardware issues lead to erroneous, compromised results, and subpar performance [1].

Big data reliability is further divided into three categories: hardware reliability, software reliability, and data reliability. The system may gather big data from a variety of sources, including sensors, IoT devices, cell phones, scanners, CVS, sensors, social media, logs, census data, RDBMS, etc. When content is ingested from a variety of sources, its accuracy and completeness are depicted as data reliability. Program failures can be caused by a misreading of the specifications, insufficient testing, a coding error, or improper software usage. The probability of errors grows in line with the exponential expansion in data volume. Therefore, in order to get accurate results, a reliability assessment of the data in use is required. Contrarily, hardware reliability results from an inadequately built system, which eventually causes performance to decline and fails to reach the required standards. Operational and architectural components make up hardware reliability. Operational reliability uses trustworthy statistical techniques to forecast equipment failure in the field and system performance using large data from modern reliability that is obtained through IoT or sensor devices attached to the system. Architectural reliability deals with the quick data retrieval from large capacity storage medium necessary for voluminous data, making it a crucial component in the successful execution of a big data project. Since data analysis is being done to support decision-making, we must employ a reliability model to cross-check for any potential flaws that could have caused the system to produce inaccurate predictions. Researchers created a variety of models to handle the reliability of big data by accounting for the external interactions of large data for precise prediction.

Han, Tian, Yoon & Lee (2012) [17] after analyzing massive amounts of data from social networks, created a big data model using Big Table and Map Reduce. Meeker and Hong in 2013[18] identified a number of applications that make use of field reliability data, such as warranty, degradation, lifetime, and recurrence field data, and investigated opportunities to assess reliable statistical techniques for predicting the performance of systems in the field. Chang Liu et al. (2014) [19] proposed the use of the Boneh-Lynn-Shacham (BLS) signature and the Multiple Huffman Table (MHT) for authorized audit in

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

a public auditing system. Tamura, Miyaoka &Yamada (2014) [20] used a three-dimensional stochastic differential equation (3D-SDE) to assess the reliability of open software systems on the cloud, assuming irregular and time-dependent fault reporting during the operating phase. The model accounts for mistakes resulting from interactions between network software, open software, and big data software as well as the software fault factor, the big data factor, and the network factor. Tamura & Yamada, (2014) [21] presented a model for 3-D stochastic differential reliability. Based on three key features of big data, parameters were evaluated using the Analytical Hierarchy approach. Kwon, Lee & Shin in 2014[22] considered that IT capabilities like data quality management and data consumption experience have a substantial impact on the desire to acquire big data analytics, and better-quality management boosts data usage regardless of its source. Li et al. (2014) [23] proposed a model to resolve conflicts between numerous big data sources with diverse structures by using an optimized framework termed CRH of two variables: truths and source reliability, where truth is defined as the value responsible for the minimum possible departure from multi-source inputs and the weights denote the degree of reliability. Tamura and Yamada (2015) [24] presented a hazard rate and clustering method for huge data situations using cloud computing that is based on SRM. They concentrated on the operation phase reliability of cloud computing with big data and created an Application for Reliability Assessment (AIR) employing cluster analysis of fault data. Tamura and Yamada (2015) [25] offered a different way to assess the dependability of cloud computing. By employing a jump-diffusion model with stochastic differential equations and two-dimensional Weiner processes, they anticipated software costs. They also define an optimal maintenance issue using a sample path while taking the level of noise into account. OSS reliability assessment (RA) methodologies were created using big data to measure component and system-level reliability. Tamura and Yamada (2015) [26] employed a hazard rate model composed of stochastic equations for RA using a data set that included cumulative faults and temporal gaps between failures. An SRM based on k-means clustering and neural networks was proposed by Tamura, Nobukawa, and Yamada, 2015 [27]. Utilizing cluster analysis findings on fault datasets gathered from databases, including Hadoop and NoSQL, and cloud applications, like Eucalyptus and OpenStack, NN was utilized to estimate cumulative faults. By integrating the human error effects with traditional PRA techniques and administrative considerations, Pence et al., 2015 [28] proposed a theoretical methodology based on Socio-technical Risk Analysis "SoTeRia" for quantifying the organizational mechanism for performance shaping factors (PSF) in human resources. A quality assessment methodology for big data was developed by Cai and Zhu in 2015 [29] using a feedback mechanism that specifies common quality components and their corresponding indicators A model for trustworthy network data mining was created by Li, He, and Ma in 2016 [30] utilizing an updated PageRank algorithm. Hu, Liu, Diao, Meng, and Sheng 2016 [31] proposed a model to leverage big data to identify the operational reliability issue of the power distribution system. The model was evaluated using NN after applying parallel index rule mining to analyze the influential aspects connected to the reliability index. A framework and model focused on research were developed by Spichkova et al. in 2016[32] employing the cloud computing platform's usability and reliability capabilities. Researchers who are exploring huge data and enormous computations can use the model. The chimney platform has undergone testing in the fields of structural biology and physics. A maintenance issue and approach were put out by Tamura and Yamada in 2017[33] to assess component reliability on the cloud platform. Through the use of a Genetic Algorithm (GA) and NN for RA, model parameters were assessed. Yan, Meng, Lu & Li (2017) [34] put forth a method as a framework for structuring large data (gathered from diverse sources with heterogeneous information) with attention to spatiotemporal factors. To make the production process clear, they simulate a variety of invisible elements, including those linked to energy conservation and preventive maintenance. In order to study the best parallel recovery strategies for replication and random and shifted 45 multi-way de-clustering data layouts, Wang, Wu, and Wang (2017) [35] built a reliability model. Nachiappan, Javadi, Calherios & Matawie (2017) [36] investigate the replication and Erasure approaches in cloud storage for massive data and list their difficulties. A text classifier was created by Xiang, Du, Ma, & Fan in 2017 [37] to assess the validity of online hotel reviews. Tamura and Yamada (2017) [38] created a deep learning-based RA model for open-source software. They created a tool that uses fault datasets to access OSS reliability. The suggested technique uses deep learning and a hazard rate model to accomplish reliability estimation. The current modeling and reliability analysis developments with reference to complicated dimension structures were explored and reviewed in 2018 by Hong, Zhang, and Meeker [39]. In 2018 Cao and Gao [40] created an SRM for big data systems utilizing fault tree analytics (FTA). FTA assesses the overall system dependability, serves as a benchmark for quality assurance, and reviews a module qualitatively to assess its likelihood of failing. Yaremchuk and Kharchenk (2018) [41] established a similarity model to spot software plagiarism and identify copies, as well as a number of other techniques and tools to spot the augmentation of program reliability. Three hybrid dependability models were created by Govindasamy and Dillibabu (2018) [3] by integrating previously existing NHPP models. With the aid of comparison

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

criteria, models were validated. In order to evaluate the parameters, MLE and GA were utilized. In 2022, Wang, Zhang, and Yang [42] proposed a RAM based on the presumption that the rate of fault introduction for Open System Software (OSS) will gradually decline. considering fault severity levels (CFSL) in OSS was the subject of a hazard rate model published by Yanagisawa, Tamura, Anand, & Yamada (2022) [43]. Their study's goal was to create a Hazard rate model for two types of fault data in the Bug Tracking System using covariate vectors and CFSL adaptive to baseline hazard function.

III. Overview of Applied Software Reliability Models and Parameter Estimation Techniques

NHPP models in three to create the hybrid model, Duane, Exponential, and PZIFD were used. The parameters were then assessed using statistical techniques MLE and LSE, and they were then further improved using soft computing techniques GA, SA, and PSO utilizing seven criteria values. The majority of SRGMS were built around a non-homogeneous poisoning process (NHPP). NHPP models are actual processes that make use of stochastic methodologies to estimate the reliability of a system utilizing appropriate interpretations from software testing and debugging. The mean value function (MVF) follows the Poisson distribution and the total number of defects is denoted by m(t). Different MVFs can be used to create various NHPP models, or the current MVF of NHPP models can be modified. Successful modeling of a software failure process is only possible after examining the testing process's variables. The MVF and intensity function (InF) of software reliability models utilized in hybrid model development are shown in Table 1 below.

Table I. NHPP models used in Hybrid model development

Models	Mean Value Function	Intensity Function
Exponential Model [47]	$m(t) = at$	$\lambda(t) = a$
Duane Model (DM) [48]	$m(t) = at^b \ \ a > 0, b > 0$	$\lambda(t) = abt^{(b-1)}$
Pham-Zhang Imperfect Fault Detection Model (PZIFD)[49]	$m(t) = a - a\,e^{-bt}\,(1 + (b+d)t + bdt^2)$	$\lambda(t) = ae^{-bt}[bt(b-d) + d(b^2t^2-1)]$

A. Parameter Estimation Techniques

The correct value of the parameters is a prerequisite for both the reliability estimation and the model validation. Large datasets and statistical methods provide a numerical estimation of the parameter values, which is then used to assess the model's goodness of fit. Two of the most used statistical techniques for parameter estimation in reliability modeling are MLE and LSE. Because of qualities like consistency, sufficiency, and parameter invariance, MLE is a favored standard approach that is frequently utilized. LSE is typically assessed using linear regression, a sum of squared errors, a root mean square deviation, and R^2 and is a viable option for linear models with medium or small samples (proportion variance). By minimizing the discrepancy between the observed and estimated values, LSE evaluates the parameters.

B. Least Square Estimation

As a result of its lower bias or faster normalcy method, the LSE excelled in predicting small data sets. The relative inaccuracy between observed and estimated values is what LSE attempts to reduce. Calculating a model's intensity function value or rate of error yields the estimated values of the model.

$$\text{LSE} = \sum_{i=1}^{n}((\lambda(t)_{Eval} - \lambda(t)_{Oval})\text{^}2 \quad (1)$$

C. Maximum Likelihood Estimation

The MLE technique selects parameter values that maximize the loglikelihood function, and the likelihood function L is obtained by utilizing the intensity function (t) as the input λ(t)

$$\text{L} = \prod_{i=1}^{n}\lambda(t_i). \quad (2)$$

and loglikelihood function is given as (Pham,2006)

$$\text{Log L} = \log\left(\prod_{i=1}^{n}\lambda(t_i)\right) \quad (3)$$

$$= \sum_{i=1}^{n}\log\left(\lambda(t_i) - m(t_n)\right). \quad (4)$$

The preceding equation is differentiated m times for each parameter with respect to time for a model with m parameters, producing m equations to solve. These equations are then solved using algorithms for "m" equations. As a result, the accuracy of the parameter now hinges on how well these approaches can avoid local minima and determine appropriate values.

A growing trend in parameter value optimization is soft computing. Numerous researchers have recently used a variety of soft computing techniques, including GA, NN, Fuzzy Logic, Support Vector Machines, Particle Swarm, Ant Colony, Gray Wolf, Cuckoo Search, Sparrow Search, and Artificial Bee Colonies, among others, for software engineering optimization

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

and reliability assessment. Soft computing techniques take advantage of uncertain, approximative, incomplete, and imprecise behavior to produce useful, economical dependable models. We utilized the techniques listed below to optimize the parameter value.

D. Genetic Algorithm

GA is a simulation of the process of species creation that is based on natural selection in biology. Chromosomes make up a population, and GA creates new populations by choosing chromosomes from the present generation and creating new ones through mutation while employing a fitness function [44]. The creation of new chromosomes continued until a halting threshold was reached. GA is irrational in nature and uses the past to solve a problem.

E. Particle Swarm Optimization

PSO is a population-based metaheuristic optimization algorithm that draws inspiration from nature. It starts out as a swarm of randomly distributed particles that represent a potential solution. Each particle calculates its position based on its velocity, prior location, and adjacent objects, which helps shape the swarm's overall behavior [16]. Every particle in the search space is an alternative solution that has fitness for the objective function. While classical PSO converges quickly, it is simple to get stuck in local minima for complex problems, which results in premature convergence. PSO is more adaptable than GA and has the ability to manage and balance both local and global search space exploration.

F. Simulated Annealing

When there are several local optima, SA is another optimization method that can be used to locate global optima. The term "annealing" is derived from thermodynamics, which describes heating and cooling metal to change its physical characteristics due to internal structural changes. Metal keeps its new structure and characteristics after cooling. Instead of material energy, SA provides the problem's objective function that needs to be maximized. To replicate the heating and cooling processes, the temperature is handled as a variable in SA. When the temperature is high, the algorithm accepts more solutions more frequently, enabling it to exit any local optimum. As the temperature drops, it becomes more likely that one will accept a suboptimal answer, compelling the algorithm to concentrate on the region of the search space where the near-optimal solution can be located. For huge, complicated problems with many local optimums, SA is an effective method that uses a slow cooling process to locate the solution that is close to the ideal.

G. Comparison Criteria

We used five comparison criteria as an objective function in GA, PSO, and SA methods to optimize the value of the parameter. These five criteria values were taken from the literature [3] and tabulated in Table II.

Table II. Comparison Criteria

Criteria	**Formula**
Mean Square Error (MSE)	$\sum_{i=1}^{n} \frac{(Oval(i) - Eval(i))\text{^}2}{n}$
Mean Absolute Deviation (MAD)	$\sum_{i=1}^{n} \frac{\lvert Oval(i) - Eval(i)\rvert}{n}$
Root Mean Square Error (RMSE)	$\sqrt{\sum_{i=1}^{n} \frac{(Oval(i) - Eval(i))\text{^}2}{n}}$
Mean Magnitude Of Relative Error (MMRE)	$\frac{1}{n}\sum_{i=1}^{n} \left\lvert \frac{Oval(i) - Eval(i)}{Oval(i)} \right\rvert$
Predictive Power (PP)	$\left(\frac{Eval(i) - Oval(i)}{Oval(i)}\right)\text{^}2$

IV. HYBRID RELIABILITY MODEL

A good model is regarded as dependable if it is straightforward, widely applicable, and successfully forecasts future failures. NHPP models are useful for figuring out a model's combined software and hardware reliability. By integrating the MVFs or InF of two (or more) NHPP models, one can create an NHPP hybrid model. The generated hybrid model's MVF is represented by the resulting combined MVF (or InF). So, by merging different NHPP models with significant parameter-set, we may create a hybrid model that provides a decent match but requires more computations and has lower dependability confidence. We created a hybrid model by merging the PZIFD model, the Duane model, and the exponential model to cover pure software faults, hardware-induced software errors, and manual-induced software bugs. Given below is the hybrid model's combined MVF and InF with its six parameters (a, b, d, x, y, and p).

$$m(t) = a - a\,e^{-bt}\,(1 + (b+d)t + bdt^2) + xt^y + pt \quad (5)$$

$$\lambda(t) = ae^{-bt}[bt(b-d) + d(b^2t^2 - 1)] + xyt^{(y-1)} + p \quad (6)$$

A. Methodology

In this study, we constructed a hybrid model and used five different approaches to identify its parameters: two statistical and three soft computing-based. To optimize the parameter value utilizing GA, SA, and PSO, we use five comparison criteria. We evaluate the intensity function using all parameter values in order to find the optimal optimizer for the

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

hybrid model. We then compute a weighted estimation function [1]. Weights ranged from 20 to 1 and were gradually reduced by 2 points using a ten-point error deviation. When there is no error and there is a 10 percent difference between the observed and experimented values, weight 20 is assigned. The following is the weighted estimation function:

$$F=20e^{0}+18e^{1}+16e^{2}+14e^{3}+12e^{4} +10e^{5}+8e^{6}+6e^{7}+4e^{8}+2e^{9}+1e^{10} \quad (6)$$

V. Experiment Work and Result Discussion

Two statistical approaches (MLE and LSE) and three soft computing techniques (GA, SA, and PSO) were used in MATLAB 2022b to evaluate the parameters of the hybrid model. To optimize the parameters, five criteria values were employed as the goal function across all of the soft computing approaches. We used four different data sets to find the optimal optimizer for the hybrid model by analyzing fault data from a large analytical system tabulated in Table III. The first three datasets (DS1, DS2, and DS3) are from a Big Data system, whereas the fourth (DS4) is a failure dataset from a medium-sized project.

Table III. Dataset Used

Dataset	Project
DS1	Software Analytical project.[7]
DS2	Fault data sets were collected from the Apache Storm project (STORM) [6].
DS3	Fault data sets collected from Apache Chemistry OpenCMIS project (OpenCMIS) [6].
DS4	data set (2008) collected during the testing process of a middle-size software project.[5]

Parameters of the hybrid model using statistical evaluation using MLE and LSE are shown below in Table IV.

Table IV. Parameters evaluated using Statistical Methods

DS1						
MLE	0.5	0.5	0.5	0.6	-0.3	-1380.9
LSQ	0.5112	0.4852	0.5075	0.7524	1.3978	2.7792
DS2						
MLE	0	0.2033	0.5161	0.0266	-6.4033	0.0303
LSQ	2.0346	0.1572	2.8603	0.1001	0.5559	5.4312
DS3						
MLE	0	0.3024	0.5191	0.0466	-7.4203	0.05
LSQ	1.9223	0.3857	1.1211	2.196	0.6543	0.1997
DS4						
MLE	0.5	0.5	0.5	0.5946	0.4035	-50.1346
LSQ	0.5261	0.3488	0.5225	2.0034	1.0563	5.723

Parameter values using GA, SA, and PSO using seven comparison values are tabulated below from Table V to Table VII for DS1 to DS4 respectively.

Table V. Parameter optimization using GA for DS1 to DS4

GA(DS1)						
MLE	4.10	0.50	24.74	12.64	-2.05	-0.02
LSE	69.07	0.13	-0.23	-861.34	0.11	12.29
MSE	0.94	0.13	9.70	16.52	0.45	8.19
MAD	5.00	8.47	33.71	0.34	1.66	3.11
RMSE	4.72	0.15	0.88	0.12	1.82	7.37
MMRE	12.54	8.99	16.53	0.46	0.21	3.97
PP	39.29	12.21	4.88	8.13	0.57	1.18
GA(DS2)						
MLE	3.01	2.76	-9.09	-9.95	-1.22	-0.09
LSE	39.53	9.06	3.25	-13.09	0.61	6.65
MSE	-2.57	5.17	-5.79	0.99	1.28	-1.61
MAD	10.00	9.99	27.97	-0.56	0.48	0.08
RMSE	-33.86	8.59	-7.70	-6.69	0.67	2.81
MMRE	5.33	-2.18	2.08	-8.78	-5.90	-0.96
PP	-7.40	-7.63	9.43	-9.26	9.67	3.02
GA(DS3)						
MLE	6.55	9.17	-9.99	2.89	0.48	-1.01
LSE	-1.66	4.46	3.90	6.54	0.13	0.77
MSE	-12.53	8.57	-16.72	10.32	-0.21	0.81
MAD	-3.28	8.14	-33.50	12.45	-0.08	0.25
RMSE	3.23	2.27	-2.29	-0.32	0.23	0.68
MMRE	4.02	0.72	-5.49	1.65	2.58	9.50
PP	-3.50	8.99	-9.56	9.02	5.01	-5.09
GA(DS4)						
MLE	-9.55	0.87	243.42	0.05	-0.07	0.04
LSE	-44.90	0.10	-1.10	-1330.40	-0.30	7.10
MSE	7.97	1.05	8.77	-6.57	-4.47	5.74
MAD	-5.71	0.06	3.95	0.61	-5.54	-4.14
RMSE	10.64	0.98	4.80	-38.35	-0.66	5.10
MMRE	9.81	1.04	7.29	-16.35	-2.05	1.76
PP	-11.92	0.97	-8.14	-17.56	-3.20	1.38

Table VI. Parameter optimization using SA for DS1 to DS4

SA(DS1)						
MLE	8.00	43.22	25.86	30.83	124.67	57.41
LSE	24.90	7.14	0.30	0.90	1.37	2.63
MSE	2.68	0.16	2.43	8.25	0.39	8.80
MAD	58.81	19.06	0.22	1.88	0.10	6.69
RMSE	111.35	46.84	35.73	0.20	1.46	9.55
MMRE	22.58	11.37	56.11	1.29	0.32	3.95
PP	2.25	49.75	10.29	14.91	0.27	2.11
SA(DS2)						
MLE	3.01	2.76	-9.09	-9.95	-1.22	-0.09
LSE	0.57	0.16	15.12	8.38	0.67	2.87
MSE	40.13	49.96	13.39	0.10	1.71	0.10
MAD	39.77	22.93	0.64	0.11	0.11	0.11
RMSE	3.36	51.89	22.13	0.27	1.49	0.14
MMRE	0.50	0.50	0.50	0.50	0.50	0.50
PP	0.50	0.50	0.50	0.50	0.50	0.50
SA(DS3)						
MLE	80.14	44.21	146.13	68.20	235.11	104.40
LSE	19.34	0.29	0.18	10.49	0.18	0.10
MSE	2.83	107.92	52.43	0.68	0.89	0.17
MAD	29.77	43.62	20.35	0.10	0.11	0.10
RMSE	0.50	0.50	0.50	0.50	0.50	0.50
MMRE	0.50	0.50	0.50	0.50	0.50	0.50

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

PP	0.50	0.50	0.50	0.50	0.50	0.50
SA(DS4)						
MLE	23.32	7.20	24.47	85.76	134.16	3.74
LSE	6.09	0.11	9.76	11.26	0.75	5.70
MSE	9.76	53.24	27.93	47.33	0.51	0.10
MAD	55.89	48.67	38.94	123.37	0.27	0.27
RMSE	37.53	55.12	66.70	52.01	0.40	3.10
MMRE	8.29	83.50	31.26	150.63	0.10	0.11
PP	29.70	58.07	4.05	72.79	0.19	0.12

Table VII. Parameter optimization using PSO for DS1 to DS4

PSO(DS1)						
MLE	419.90	1000.00	786.20	475.30	119.40	0.10
LSE	813.10	300.75	751.05	0.10	0.10	9.63
MSE	945.43	219.03	928.80	0.10	0.10	9.63
MAD	929.80	1000.00	988.60	0.10	0.10	6.00
RMSE	108.84	659.51	951.67	192.33	0.10	0.10
MMRE	362.99	801.54	443.62	22.21	0.10	3.78
PP	410.46	214.89	455.85	41.01	0.10	2.30
PSO(DS2)						
MLE	3.01	2.76	-9.09	-9.95	-1.22	-0.09
LSE	60.90	0.10	0.10	61.37	0.10	0.10
MSE	203.82	977.47	612.44	11.70	0.10	0.10
MAD	607.72	16.57	88.70	0.10	0.10	0.10
RMSE	0.10	0.10	41.31	71.11	0.10	0.10
MMRE	992.05	594.74	897.44	501.57	415.51	500.68
PP	797.76	78.95	529.52	99.82	117.05	610.10
PSO(DS3)						
MLE	1000.00	1000.00	531.80	632.00	230.40	1000.00
LSE	193.87	452.40	432.44	0.10	0.10	0.64
MSE	974.94	652.89	0.33	12.85	0.10	0.10
MAD	0.76	8.72	107.78	0.10	0.10	0.10
RMSE	1000.00	984.10	0.10	12.90	0.10	0.10
MMRE	510.90	4.44	925.50	757.34	118.85	827.10
PP	575.04	738.63	596.80	254.29	965.39	63.06
PSO(DS4)						
MLE	0.10	966.20	835.60	1000.00	112.70	952.90
LSE	475.60	0.10	0.10	0.10	0.10	0.10
MSE	611.16	464.80	970.08	191.25	0.10	5.31
MAD	290.37	471.35	539.47	341.57	0.10	1.70
RMSE	857.80	981.00	49.85	191.25	0.10	5.31
MMRE	824.46	959.24	468.38	150.92	0.10	0.10
PP	973.01	998.95	750.69	176.92	0.10	0.10

A Comparison between Evaluation methods for all datasets is carried out to determine the good performance of a method across all datasets as shown below in Fig. 1.

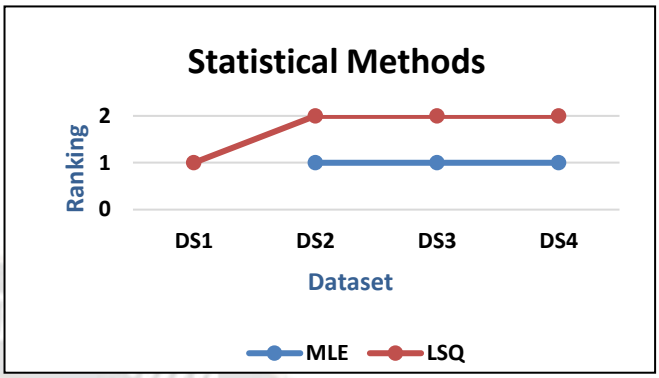

Figure 1. Comparison between statistical methods.

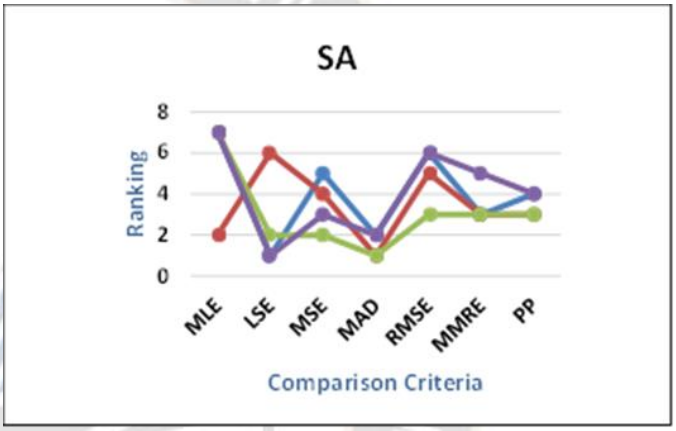

Figure 2. Ranking of GA values evaluated using comparision criteria

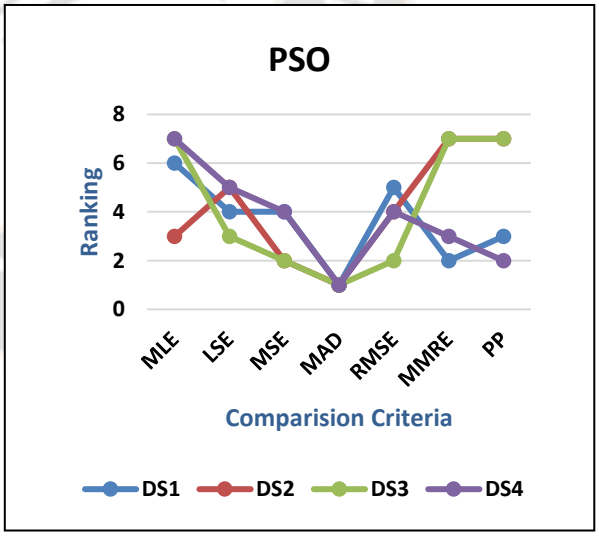

Figure 3. Ranking of PSO values evaluated using comparision criteria

It is clear from the Fig. 1, that between MLE and LSE MLE gives good results as compared to LSE for datasets DS2, DS3, and DS4. Whereas LSE is better than MLE for DS1. Fig. 2 and Fig 3. show the performance of GA and PSO for all datasets and

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

GA gives better performance for DS1 while SA performs better for DS3.

We also determine the performance of each method for individual datasets to compare which criteria when used as an objective function is giving the best result. Also calculated is the highest value of the estimation function for each evaluation and method and compared to determine the method giving accurate predictions. The results of the comparisons are shown in the charts of Fig. 4. Following observations were made regarding estimation function values evaluated using various methods for all datasets.

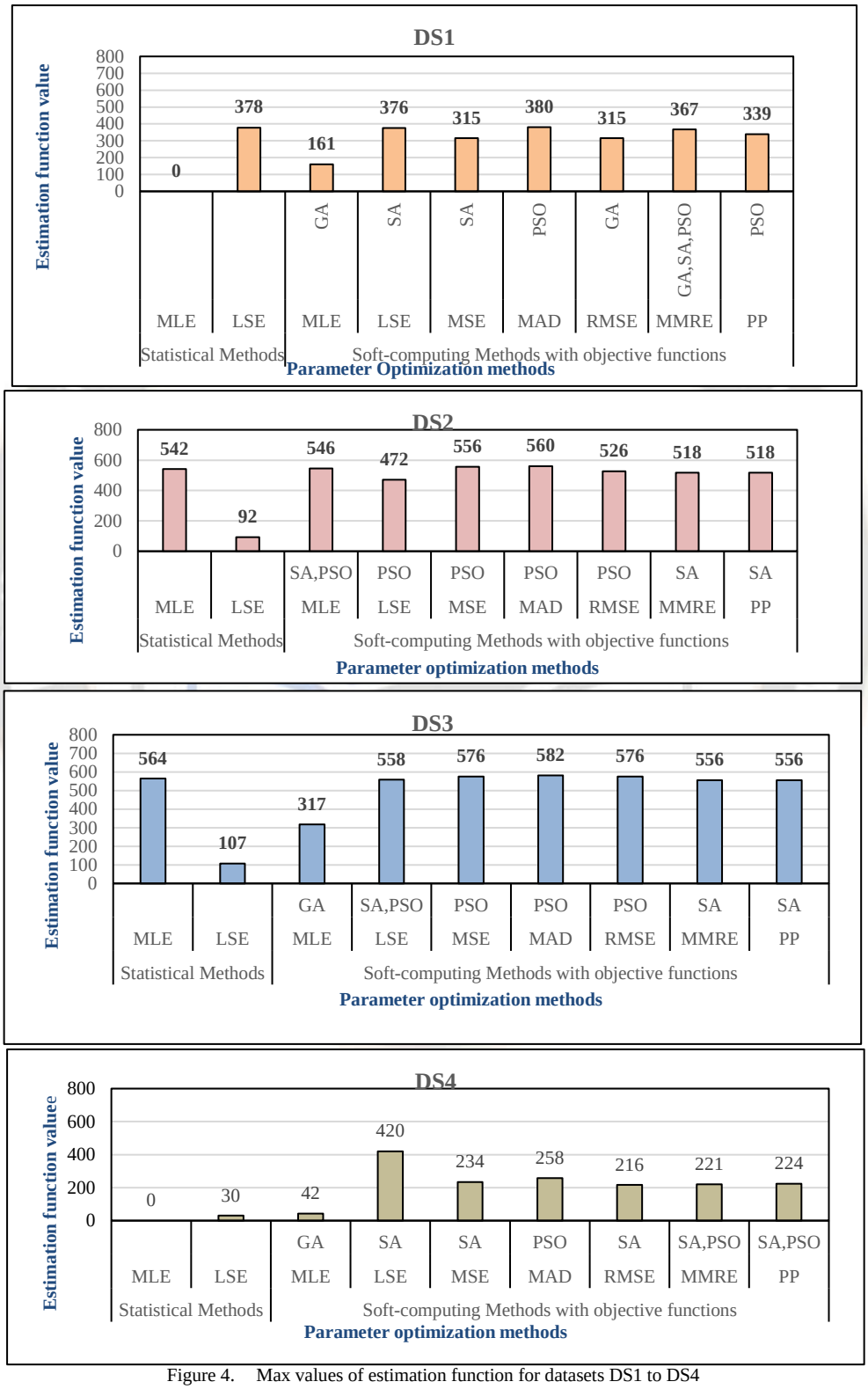

Figure 4. Max values of estimation function for datasets DS1 to DS4

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

DS1: Figure demonstrates that the PSO using MAD as an objective function gives the highest value of the estimation function giving more accurate prediction as compared to other discussed methods. The second-best prediction is given by LSE using the "Levenberg-Marquardt" algorithm for optimization.
DS2: PSO is the best evaluator for DS2 having both the first and second-best estimation function score using MAD and MSE respectively.
DS3: Again, PSO is a better evaluator than other methods for DS3 Giving both the first and second highest estimation function value for MAD and (MSE, RMSE) respectively.
DS4: SA gives way better performance than any other evaluator for DS4 which is not of fault data of a big data system. PSO with MAD has the second-highest estimation function score.

When we consider all method's highest estimation scores as shown in Fig. 5, we find that PSO using MAD as an objective function gives the best performance for all big datasets. So, we can safely say according to the depicted results that PSO is the best method to be used for the big data hybrid reliability model. Moreover, if we consider the individual performance of each method based on their estimation accuracy, we can infer that the estimation capability of MLE is better for DS3 as compared to other datasets. LSE gives the best prediction for DS1 compared to DS2, DS3, and DS4. Giving first and second highest estimation accuracy is PSO using MAD, MSE, and RMSE for DS3. The third position is bagged by statistical MLE using the "trust-region dogleg algorithm" as an optimizer. SA and PSO both are at position four for DS3 using LSE as an objective function.SA with MMRE and PP for DS3 is at number five and MLE as an objective function for both PSO and SA gets the last position.

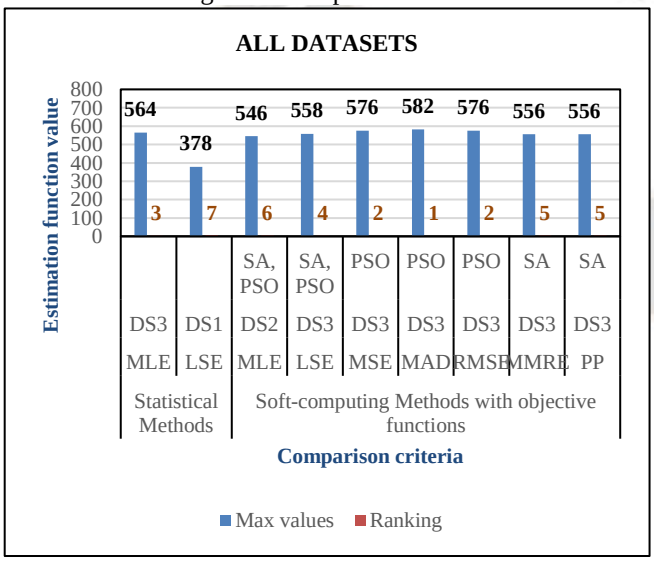

Figure 5. Max values of estimation function in considered methods.

The model is validated by comparing it to two other models Yamada Delayed and Yamada imperfect debugging model number 1. Shown below in Fig. 6 are the intensity function plot of our developed Hybrid model (HM) with these traditional models for datasets DS1, DS2, DS3, and DS4.

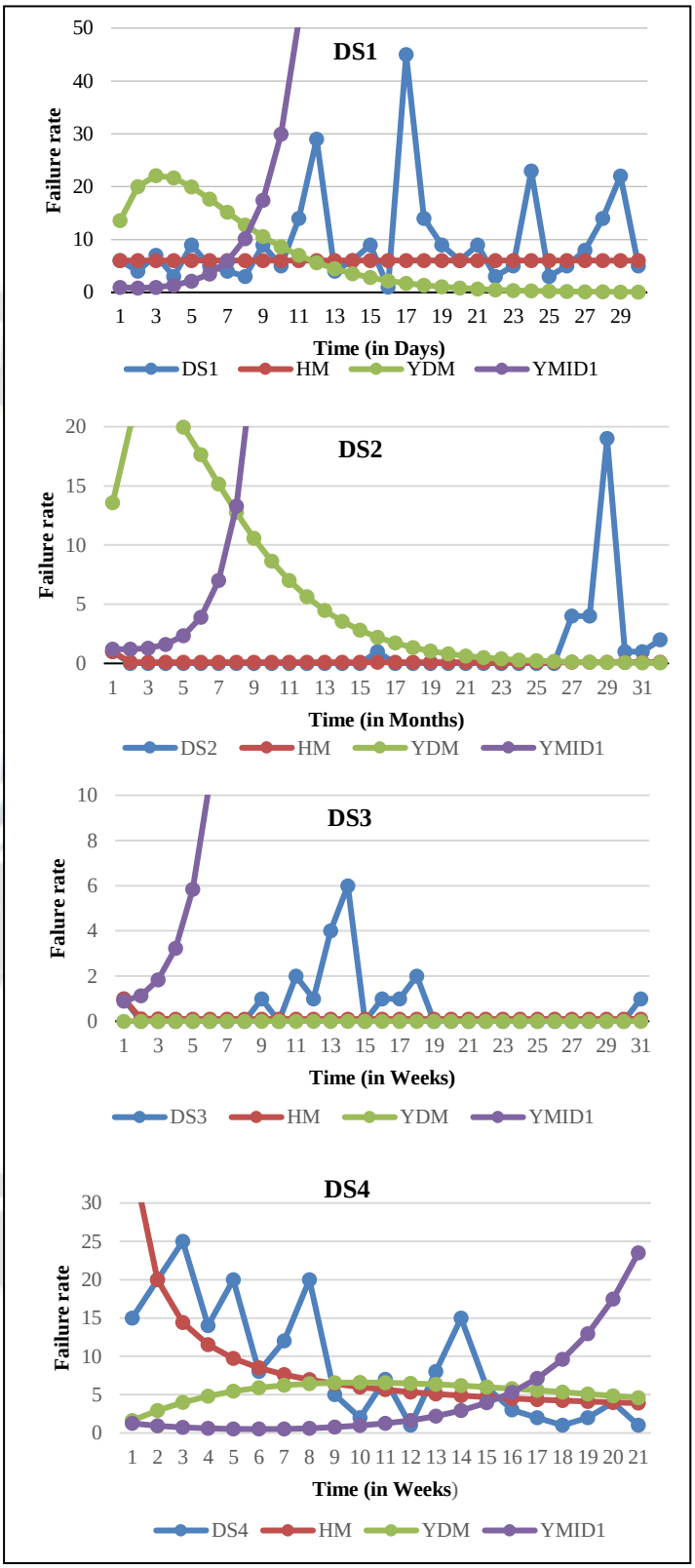

Figure 6. Intensity function comparison of hybrid Model, YDM and YDIM1.

It is evident from the graphs that the intensity function plot of the hybrid model gives better prediction than the other two

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

well-known traditional models for all datasets. Thus, we can safely say that the developed hybrid model is a better predictor than traditional models for big data systems.

VI. CONCLUSION:

The study presents a hybrid model that was built to forecast the dependability of software. The parameters of the model were determined with the help of four datasets through the use of a variety of statistical and soft computing approaches. To optimize the parameter values, a total of five different criteria value formulas, together with LSE and MLE, were used as objective functions. There was a total of four datasets utilized in the process of determining the best optimizer with criteria value, and it was discovered that PSO with MAD is the best estimator for DS1, DS2, and DS3, while DS4 received second place for its performance. Based on the results of the experiment, we are able to draw the conclusion that PSO is the technique that is most suited for hybrid models that deal with Big Data analytical systems and uses MAD as its objective function for predictions. Moreover, the comparison of the developed model with traditional models confirms its claim of better predictor than existing models for big fault data.

REFERENCES

[1]. S. Sharma, N. Kumar, and K. S. Kaswan, "Ranking of Reliability Models based on Accurate Estimation and Weighted Function," *Proc. - 2021 3rd Int. Conf. Adv. Comput. Commun. Control Networking, ICAC3N 2021*, pp. 1679–1685, 2021, doi: 10.1109/ICAC3N53548.2021.9725534.

[2]. A. L. Goel and K. Okumoto, "Time-Dependent Error-Detection Rate Model for Software Reliability and Other Performance Measures," *IEEE Trans. Reliab.*, vol. R-28, no. 3, pp. 206–211, 1979, doi: 10.1109/TR.1979.5220566.

[3]. P. Govindasamy and R. Dillibabu, "Development of software reliability models using a hybrid approach and validation of the proposed models using big data",*J Supercomput*, Springer vol. 76, no. 4, 2020.

[4]. S. Osaki, "S-Shaped Software Reliability Growth Models and Their Applications," *IEEE Trans. Reliab.*, vol. R-33, no. 4, pp. 289–292, 1984, doi: 10.1109/TR.1984.5221826.

[5]. S. Yamada, M. Ohba, and S. Osaki, "S-Shaped Reliability Growth Modeling for Software Error Detection," *IEEE Trans. Reliab.*, vol. R-32, no. 5, pp. 475–484, 1983, doi: 10.1109/TR.1983.5221735.

[6]. S. Yamada and S. Osaki, "Software Reliability Growth Modeling: Models and Applications," *IEEE Trans. Softw. Eng.*, vol. SE-11, no. 12, pp. 1431–1437, 1985, doi: 10.1109/TSE.1985.232179.

[7]. J. D. Musa and K. Okumoto, "A logarithmic Poisson execution time model for software reliability measurement". *In Proceedings of the 7th international conference on Software engineering*,pp. 230-238, 1984.

[8]. A. Wood, "Software-reliability growth model: primary-failures generate secondary-faults under imperfect debugging and . IEEE Transactions on Reliability, 43, 3, 408 (September 1994)," *Microelectron. Reliab.*, vol. 36, no. September, p. 446, 1996, Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/0026271496819585.

[9]. K. Ohishi, H. Okamura, and T. Dohi, "Gompertz software reliability model: Estimation algorithm and empirical validation," *J. Syst. Softw.*, vol. 82, no. 3, pp. 535–543, 2009, doi: 10.1016/j.jss.2008.11.840.

[10]. S. Yamada, J. Hishitani, and S. Osaki, "Software-Reliability Growth with a Weibull Test-Effort: A Model & Application," *IEEE Trans. Reliab.*, vol. 42, no. 1, pp. 100–106, 1993, doi: 10.1109/24.210278.

[11]. S. Yamada, K. Tokuno, and S. Osaki, "Imperfect debugging models with fault introduction rate for software reliability assessment," *Int. J. Syst. Sci.*, vol. 23, no. 12, pp. 2241–2252, 1992, doi: 10.1080/00207729208949452.

[12]. A. Iannino and J. D. Musa, *Software Reliability*, vol. 30, no. C. 1990.

[13]. C. Y. Huang, M. R. Lyu, and S. Y. Kuo, "A unified scheme of some nonhomogeneous poisson process models for software reliability estimation," *IEEE Trans. Softw. Eng.*, vol. 29, no. 3, pp. 261–269, 2003, doi: 10.1109/TSE.2003.1183936.

[14]. H. Pham, L. Nordmann, and X. Zhang, "A general imperfect-software-debugging model with s-shaped fault-detection rate," *IEEE Trans. Reliab.*, vol. 48, no. 2, pp. 169–175, 1999, doi: 10.1109/24.784276.

[15]. S. Yamada, H. Ohtera, and H. Narihisa, "Software Reliability Growth Models with Testing-Effort," *IEEE Trans. Reliab.*, vol. 35, no. 1, pp. 19–23, 1986, doi: 10.1109/TR.1986.4335332.

[16]. C. Jin and S. W. Jin, "Parameter optimization of software reliability growth model with S-shaped testing-effort function using improved swarm intelligent optimization," *Appl. Soft Comput. J.*, vol. 40, pp. 283–291, 2016, doi: 10.1016/j.asoc.2015.11.041.

[17]. 17] X. Han, L. Tian, M. Yoon, and M. Lee, "A big data model supporting information recommendation in social networks," *Proc. - 2nd Int. Conf. Cloud Green Comput. 2nd Int. Conf. Soc. Comput. Its Appl. CGC/SCA 2012*, pp. 810–813, 2012, doi: 10.1109/CGC.2012.125.

[18]. Y. Hong, M. Zhang, and W. Q. Meeker, "Big data and reliability applications: The complexity dimension," *J. Qual. Technol.*, vol. 50, no. 2, pp. 135–149, 2018, doi: 10.1080/00224065.2018.1438007.

[19]. C. Liu *et al.*, "Authorized public auditing of dynamic big data storage on cloud with efficient verifiable fine-grained updates," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 9, pp. 2234–2244, 2014, doi: 10.1109/TPDS.2013.191.

[20]. Y. Tamura, K. Miyaoka and S. Yamada, "Reliability analysis based on three-dimensional stochastic differential equation for big data on cloud computing," *2014 IEEE International Conference on Industrial Engineering and Engineering Management*, 2014, pp. 863-867, doi: 10.1109/IEEM.2014.7058761.

[21]. Y. Tamura and S. Yamada, "Reliability Analysis Based on AHP and Software Reliability Models for Big Data on Cloud Computing," vol. 1, no. 1, pp. 43–49, 2014.

[22]. Q. Li, Y. Li, J. Gao, B. Zhao, W. Fan, and J. Han, "Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation," *Proc. ACM SIGMOD Int. Conf. Manag. Data*, pp. 1187–1198, 2014, doi: 10.1145/2588555.2610509.

[23]. O. Kwon, N. Lee, and B. Shin, "Data quality management, data usage experience and acquisition intention of big data analytics," *Int. J. Inf. Manage.*, vol. 34, no. 3, pp. 387–394, 2014, doi: 10.1016/j.ijinfomgt.2014.02.002.

[24]. Y. Tamura and S. Yamada, "Software reliability assessment tool based on fault data clustering and hazard rate model considering cloud computing with big data," *2015 4th Int. Conf. Reliab. Infocom Technol. Optim. Trends Futur. Dir. ICRITO 2015*, vol. d, pp. 1–6, 2015, doi: 10.1109/ICRITO.2015.7359208.

[25]. Y. Tamura and S. Yamada, "Reliability analysis based on a jump diffusion model with two wiener processes for cloud computing with big data," *Entropy*, vol. 17, no. 7, pp. 4533–4546, 2015, doi: 10.3390/e17074533.

[26]. Y. Tamura and S. Yamada, "Software reliability analysis considering the fault detection trends for big data on cloud computing," *Lect. Notes Electr. Eng.*, vol. 349, pp. 1021–1030, 2015, doi: 10.1007/978-3-662-47200-2_106.

[27]. Y. Tamura, Y. Nobukawa, and S. Yamada, "A method of reliability assessment based on neural network and fault data clustering for cloud with big data," *2015 IEEE 2nd Int. Conf. InformationScience Secur. ICISS 2015*, pp. 4–7, 2016, doi: 10.1109/ICISSEC.2015.7370965.

[28]. J. Pence *et al.*, "Quantifying organizational factors in human reliability analysis using the big data-theoretic algorithm," *Int. Top. Meet. Probabilistic Saf. Assess. Anal. PSA 2015*, vol. 2, pp. 650–659, 2015.

[29]. L. Cai and Y. Zhu, "The challenges of data quality and data quality assessment in the big data era," *Data Sci. J.*, vol. 14, pp. 1–10, 2015, doi: 10.5334/dsj-2015-002.

[30]. J. Li, Y. He, and Y. Ma, "Research of network data mining based on reliability source under big data environment," *Neural Comput. Appl.*, vol. 28, pp. 327–335, 2017, doi: 10.1007/s00521-016-2349-x.

[31]. L. Hu, K. Y. Liu, Y. Diao, X. Meng, and W. Sheng, "Operational reliability evaluation method based on big data technology," *Proc. - 2016 Int. Conf. Cyber-Enabled Distrib. Comput. Knowl. Discov. CyberC 2016*, pp. 341–344, 2017, doi: 10.1109/CyberC.2016.71.

[32]. M. Spichkova, H. W. Schmidt, I. I. Yusuf, I. E. Thomas, S. Androulakis, and G. R. Meyer, "Towards modelling and implementation of reliability and usability features for research-oriented cloud computing platforms," *Commun. Comput. Inf. Sci.*, vol. 703, pp. 158–178, 2016, doi: 10.1007/978-3-319-56390-9_8.

[33]. Y. Tamura, T. Takeuchi, and S. Yamada, "Software Reliability and Cost Analysis Considering Service User for Cloud with Big Data," *Int. J. Reliab. Qual. Saf. Eng.*, vol. 24, no. 2, pp. 1–14, 2017, doi: 10.1142/S0218539317500097.

[34]. J. Yan, Y. Meng, L. Lu, and L. Li, "Industrial Big Data in an Industry 4.0 Environment: Challenges, Schemes, and Applications for Predictive Maintenance," *IEEE Access*, vol. 5, no. c, pp. 23484–23491, 2017, doi: 10.1109/ACCESS.2017.2765544.

[35]. J. Wang, H. Wu, and R. Wang, "A new reliability model in replication-based big data storage systems," *J. Parallel Distrib. Comput.*, vol. 108, pp. 14–27, 2017, doi: 10.1016/j.jpdc.2017.02.001.

[36]. R. Nachiappan, B. Javadi, R. N. Calheiros, and K. M. Matawie, "Cloud storage reliability for Big Data applications: A state of the art survey," *J. Netw. Comput. Appl.*, vol. 97, pp. 35–47, 2017, doi: 10.1016/j.jnca.2017.08.011.

[37]. Z. Xiang, Q. Du, Y. Ma, and W. Fan, "Assessing reliability of social media data: lessons from mining TripAdvisor hotel reviews," *Inf. Technol. Tour.*, vol. 18, no. 1–4, pp. 43–59, 2018, doi: 10.1007/s40558-017-0098-z.

[38]. Y. Tamura and S. Yamada, "Fault Identification and Reliability Assessment Tool Based on Deep Learning for Fault Big Data," *Softw. Netw.*, vol. 2017, no. 1, pp. 161–167, 2017, doi: 10.13052/jsn2445-9739.2017.008.

[39]. Y. Hong, M. Zhang, and W. Q. Meeker, "Big data and reliability applications: The complexity dimension," *J. Qual. Technol.*, vol. 50, no. 2, pp. 135–149, 2018, doi: 10.1080/00224065.2018.1438007.

[40]. R. Cao and J. Gao, "Research on reliability evaluation of big data system," *2018 3rd IEEE Int. Conf. Cloud Comput. Big Data Anal. ICCCBDA 2018*, pp. 261–265, 2018, doi: 10.1109/ICCCBDA.2018.8386523.

[41]. S. Yaremchuk and V. Kharchenko, "Big data and similarity-based software reliability assessment: The technique and applied tools," *Proc. 2018 IEEE 9th Int. Conf. Dependable Syst. Serv. Technol. DESSERT 2018*, no. May, pp. 485–490, 2018, doi: 10.1109/DESSERT.2018.8409182.

[42]. J. Wang, C. Zhang, and J. Yang, "Software reliability model of open source software based on the decreasing trend of fault introduction," *PLoS One*, vol. 17, no. 5 May, pp. 1–18, 2022, doi: 10.1371/journal.pone.0267171.

[43]. T. Yanagisawa, Y. Tamura, A. Anand, and S. Yamada, "A Software Reliability Model for OSS Including Various Fault Data Based on Proportional Hazard-Rate Model," *Am. J. Oper. Res.*, vol. 12, no. 01, pp. 1–10, 2022, doi: 10.4236/ajor.2022.121001.

[44]. T. Minohara and Y. Tohma, "Parameter estimation of hyper-geometric distribution software reliability growth model by genetic algorithms," *Proc. Int. Symp. Softw. Reliab. Eng. ISSRE*, no. i, pp. 324–329, 1995, doi: 10.1109/issre.1995.497673.

[45]. A. Choudhary, A. S. Baghel and O. P. Sangwan, "An efficient parameter estimation of software reliability growth models using gravitational search algorithm". *International Journal of System Assurance Engineering and Management*, *8*(1), 79-88,2017.

[46]. S. Sharma, N. Kumar, and K. S. Kaswan, "Big data reliability: A critical review," *J. Intell. Fuzzy Syst.*, vol. 40, no. 3, pp. 5501–5516, 2021, doi: 10.3233/JIFS-202503.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 1
DOI: https://doi.org/10.17762/ijritcc.v11i1.5984
***Article Received**: 08 November 2022 **Revised**: 10 December 2022 **Accepted**: 02 January 2023*

[47]. R. Miller, Exponential order statistic models of software reliability growth, IEEE Transactions on Software Engineering SE-12(1) (1986), 12–24.
[48]. M. Xie, "Software Reliability Models for Practical Applications," pp. 211–214, 1995, doi: 10.1007/978-0-387-34848-3_32.
[49]. H. Pham," System *software reliability*", Springer Science & Business Media,2007.