An Assessment of Some Entropy Measures in Predicting Bugs of Open-Source Software

Vijay Kumar, H. D. Arora and Ramita Sahni

Abstract In software, source code changes are expected to occur. In order to meet the enormous requirements of the users, source codes are frequently modified. The maintenance task is highly complicated if the changes due to bug repair, enhancement, and addition of new features are not reported carefully. In this paper, concurrent versions system (CVS) repository (http://bugzilla.mozilla.org) is taken into consideration for recording bugs. These observed bugs are collected from some subcomponents of Mozilla open-source software. As entropy is helpful in studying the code change process, and various entropies, namely Shannon, Renyi, and Tsallis entropies, have been evaluated using these observed bugs. By applying simple linear regression (SLR) technique, the bugs which are yet to come in future are predicted based on current year entropy measures and the observed bugs. Performance has been measured using various R^2 statistics. In addition to this, ANOVA and Tukey test have been applied to statistically validate various entropy measures.

Keywords Bug prediction · Entropy · Coding · Software repositories
Open-source software

V. Kumar
Department of Mathematics, Amity School of Engineering and Technology,
New Delhi 110061, India
e-mail: vijay_parashar@yahoo.com

H. D. Arora · R. Sahni (✉)
Department of Applied Mathematics, Amity Institute of Applied Sciences,
Amity University, Sector-125, Noida, Uttar Pradesh, India
e-mail: smiles_ramita@yahoo.co.in

H. D. Arora
e-mail: hdarora@amity.edu

1 Introduction

The software industry people are gaining a lot of attention due to the success and development of open-source software community. Open-source software is the software whose source code is available for modification by everyone with no central control. Due to factors like fewer bugs, better reliability, no vendor dependence, educational support, shorter development cycles, open-source software system gives an aggressive competition to the closed-source software. Due to increasing popularity of open-source software, changes in source code are unavoidable. It is a necessity to record changes properly in storage locations, viz. source code repositories. There is a direct correlation between the number of changes and faults or bugs in the software system. Bugs may be introduced at any phase of the software development life cycle, and by lying dormant in the software, bugs affect the quality and reliability of the software system, thereby making the system complex. Measuring complexity is an essential task which starts from development of code to maintenance. Entropy, a central concept of information theory, is defined as measure of randomness/uncertainty/complexity of code change. It tells us how much information is present in an event. Information theory is a probabilistic approach dealing with assessing and defining the amount of information contained in a message. While dealing with real-world problems, we cannot avoid uncertainty. The paramount goal of information theory is to capture or reduce this uncertainty.

In this paper, the data has been collected for 12 subcomponents of Mozilla open-source system, namely Doctor, Elmo, AUS, DMD, Bonsai, Bouncer, String, Layout Images, Toolbars and Toolbar Customization, Identity, Graph Server, and Telemetry Server. Initially, the number of bugs/faults present in each component is reported for 7 years from 2008 to 2014. Thereafter, for the data extracted from these subcomponents, Shannon entropy [1], Renyi entropy [2], and Tsallis entropy [3] have been evaluated for each time period, i.e., from 2008 to 2014. Simple linear regression (SLR) using Statistical Package for Social Sciences (SPSS) has been applied between the entropy calculated and observed bugs for each time period to obtain the regression coefficients. These regression coefficients have been used to calculate the predicted bugs for the coming year based on the entropy of the current year. Performance has been measured using goodness of fit curve and other R^2 statistics. In addition to this, ANOVA and Tukey test have been applied to statistically validate various entropy measures. There are many measures of entropy, but only these three entropies are considered for this study to compile and conclude results based on these measures, other measures may also be taken for further study and analysis, and comparative study is another area of research. The paper is further divided into the following sections. Section 2 contains the literature review of the work previously been done. Section 3 provides the basics of entropy measures and the code change process. Section 4 discusses the methodology adopted in this paper with data collection and preprocessing and calculation of entropy measures. In Sect. 5, the bug prediction modeling approach is described. In Sect. 6, the assessment of entropy measures has been discussed. Finally, the paper is concluded with limitations and future scope in Sect. 7.

2 Literature Review

Researchers have proposed and implemented a huge number of approaches for prediction of dormant bugs lying in the software and for measuring the reliability growth of the software (Goel and Okumoto [4]; Huang et al. [5]; Kapur and Garg [6]; Kapur et al. [7]; Kapur et al. [8]; etc.). Hassan [9] proposed information theory to measure the amount of uncertainty or entropy of the distribution to quantify the code change process in terms of entropy and used entropy-based measures to predict the bugs based on past defects. Ambros and Robbes [10] provided an extensive comparison of well-known bug prediction approaches. Singh and Chaturvedi [11] developed a theoretical agenda for developing bug tracking and reliability assessment (BTRAS) based on different classification criteria. Khatri et al. [12] presented a generalized model to find out the proportion of bug complexity present in the software for providing better software production. Singh and Chaturvedi [13] collected the source code change data of Mozilla components to validate the proposed method and to predict the bugs yet to come in future based on the current year entropy by applying simple linear regression and support vector regression techniques. Chaturvedi et al. [14] presented a novel approach to predict the potential complexity of code changes using entropy-based measures. These predicted changes help in determining the remaining code changes yet to be diffused in the software. The diffusion of change complexity has been validated using some subcomponents of Mozilla project. Singh et al. [15] proposed three approaches, namely software reliability growth models, potential complexity of code change-based models, and code change complexity-based models to determine the presence of potential bugs in the software. Sharma et al. [16] developed prediction models for determining severity level of a reported bug based on attributes, namely priority, number of comments, number of dependents, number of duplicates, complexity, summary weight, and CC list (a list of people who get mail when the bug changes) in cross-project context. They also considered bug reports of some products of Mozilla open-source project for empirical validation.

3 Entropy Measures and Code Change Process

The concept of entropy in information theory was developed by Shannon [1]. He defined a formal measure of entropy, called Shannon entropy:

$$S = -\sum_{i=1}^{n} p_i \log_2 p_i \tag{1}$$

where p_i is the probability of occurrence of an event.

A systematic approach to develop a generalization of Shannon entropy [1] was made by Renyi [2] who characterized entropy of order α defined as follows:

$$R = \frac{1}{1-\alpha} \log\left(\sum_{i=1}^{n} p_i^{\alpha}\right), \qquad \alpha \neq 1, \alpha > 0 \tag{2}$$

where α is a real parameter.

Tsallis [3] proposed another generalization of Shannon entropy [13] defined as:

$$T = \frac{1}{\alpha-1}\left(1 - \sum_{i=1}^{n} p_i^{\alpha}\right), \qquad \alpha \neq 1, \alpha > 0 \tag{3}$$

Renyi entropy [2] and Tsallis entropy [3] reduce to Shannon entropy [1] when $\alpha \to 1$.

For Renyi [2] and Tsallis entropies [3], any value of $\alpha > 0$ can be taken, other than 1 to study the variation and effect of varying alpha on entropies. So here, five values of parameter α i.e., 0.1, 0.3, 0.5, 0.7, and 0.9 are taken into consideration.

The code change process refers to study the patterns of source code changes/ modifications. Bug repair, feature enhancement, and the addition of new features cause these changes/modifications. The entropy-based estimation plays a vital role in studying the code change process. Entropy is determined based on the number of changes in a file for a particular time period with respect to total number of changes in all files. Keeping in mind the frequency of changes in the code, we can decide the specific duration to be day, month, year, etc. For example, consider that there are 13 changes occurred for four files and three periods. Let P1, P2, P3, and P4 be the four files and S1, S2, and S3 be the three periods. In S1, files P1, P2, and P4 have one change each and P3 has two changes. Table 1 depicts the total number of changes occurring in each file in respective time periods S1, S2, and S3.

Table 1 Number of changes (denoted by *) in files with respect to a specific period of time where P1, P2, P3, and P4 represent the files and S1, S2, and S3 represent the time periods

File/time	S1	S2	S3
P1	*	*	*
P2	*		*
P3	**	**	
P4	*	**	*

Total files in S1 = 5. Thus, probability of P1 for S1 = 1/5 = 0.2; probability of P2 for S1 = 1/5 = 0.2; probability of P3 for S1 = 2/5 = 0.4; and probability of P4 for S1 = 1/5 = 0.2. Similarly, we can find out the probabilities of the time periods S2 and S3. Based upon the above probabilities, we have calculated Shannon entropy [13], Renyi entropy [11], and Tsallis entropy.

4 Methodology

In this paper, first the data is collected and preprocessed and then entropy measures have been calculated.

4.1 Data Collection and Preprocessing

Mozilla is open-source software that offers choice to the users and drives innovation on the Web. It is a free software community which produces a large number of projects like Thunderbird, the bug tracking system Bugzilla, etc. In this paper, we have selected few components of Mozilla software with respective bugs from the CVS repository: http://bugzilla.mozilla.org [17]. Steps for data collection, extraction, and prediction of bugs are as follows:

1. Choose the project, select the subsystems, and browse CVS logs.
2. Collect bug reports of all subsystems, extract bugs from these reports, and arrange bugs on yearly basis for each subsystem.
3. Calculate Shannon entropy, Renyi entropy, and Tsallis entropy for each time period using these bugs reported for each subsystem.
4. Use SLR model to predict bugs for the coming year based on each entropy calculated for each time period.

In our study, we have taken a fixed period as 1 year taken from 2008 to 2014. We have considered 12 subsystems with number of bugs as 6 bugs in Doctor, 32 bugs in Elmo, 72 bugs in Graph Server, 43 bugs in Bonsai, 45 bugs in Bouncer, 33 bugs in String, 12 bugs in DMD, 90 bugs in Layout Images, 23 bugs in AUS, 39 bugs in Identity, 12 bugs in Telemetry Server, and 36 bugs in Toolbars and Toolbar Customization.

4.2 Evaluation of Shannon, Renyi, and Tsallis Entropies

This data information has been used to calculate the probability of each component for the seven time periods from 2008 to 2014 as discussed in Sect. 3. Using these probabilities Shannon [1], Renyi [2] and Tsallis entropies [3] are calculated using

Eq. (1)–(3), respectively, for each time period. For Renyi [2] and Tsallis entropies [3], five values of α, i.e., 0.1, 0.3, 0.5, 0.7, and 0.9 are considered. Table 2 shown below depicts the Shannon entropy [1], Renyi entropy [2], and Tsallis entropy [3] for each year.

From this analysis, it has been observed that Shannon entropy [1] lies between 2 and 4. It is maximum in the year 2014 and minimum in the year 2009. Renyi entropy [2] and Tsallis entropy [3] decrease as the value of α increases. At $\alpha = 0.1$, Renyi entropy [2] is maximum for each time period, and at $\alpha = 0.9$, Renyi entropy [11] is minimum for each time period. Similarly, at $\alpha = 0.1$, Tsallis entropy [3] is maximum for each time period, and at $\alpha = 0.9$, Tsallis entropy [3] is minimum for each time period.

5 Bug Prediction Modeling

Simple linear regression (SLR) [18] model is the most elementary model involving two variables in which one variable is predicted by another variable. The variable to be predicted is called the dependent variable, and the predictor is called the independent variable The SLR has been widely used to repress the dependent variable Y using independent variable X with the following equation

$$Y = A + BX \tag{4}$$

where A and B are regression coefficients.

The regression coefficients can be obtained using the simple linear regression technique using Statistical Package for Social Sciences (SPSS). After estimating the regression coefficients using different measures of entropy and historical data of observed bugs, a system is constructed with which we can predict bugs likely to occur in the next year. In this study, X, i.e., entropy measure, is considered to be an independent variable and Y, i.e., predicted bugs likely to occur next year, is considered to be a dependent variable. Here X, i.e., entropy measure, is different in each case, i.e., as Shannon entropy [1], Renyi entropy [2], and Tsallis entropy [3]. For Renyi [2] and Tsallis entropies [3], five parameter values are considered, viz. 0.1, 0.3, 0.5, 0.7, and 0.9. The following notations are used for simplicity:

X: entropy measures; Y_{o}: observed bugs; Y: predicted bugs for Shannon entropy.$\mathrm{YR}_{\propto}$ represents the predicted bugs for Renyi entropy with α varying as 0.1, 0.3, 0.5, 0.7, and 0.9, respectively.$\mathrm{YT}_{\propto}$ represents the predicted bugs for Tsallis entropy with α varying as 0.1, 0.3, 0.5, 0.7, and 0.9, respectively. Table 3 depicts the predicted bugs along with Shannon entropy [13], Renyi entropy [11], and Tsallis entropy [19].

Table 2 Shannon entropy, Renyi entropy, and Tsallis entropy for each time period with different values of α

Year	S	$R(0.1)$	$R(0.3)$	$R(0.5)$	$R(0.7)$	$R(0.9)$	$T(0.1)$	$T(0.3)$	$T(0.5)$	$T(0.7)$	$T(0.9)$
2008	2.4909	0.8342	0.8131	0.7935	0.7799	0.7578	5.1493	3.8698	2.9865	2.3599	1.9065
2009	2.4761	0.7745	0.7673	0.7609	0.7544	0.7484	4.4203	3.4929	2.8026	2.2798	1.8805
2010	2.7319	0.8928	0.8735	0.8568	0.8416	0.8284	5.9567	4.4119	3.3632	2.6284	2.1016
2011	2.7281	0.8955	0.8798	0.8639	0.8471	0.8299	5.9973	4.4714	3.4075	2.6509	2.1058
2012	2.9942	0.9873	0.9638	0.9435	0.9251	0.9088	7.4868	5.3278	3.9264	2.9821	2.3277
2013	3.0171	1.0599	1.0222	0.9870	0.9535	0.9227	8.8831	5.9949	4.2305	3.1073	2.3671
2014	3.1596	1.0309	1.0108	0.9924	0.9749	0.9588	8.3007	5.8589	4.2697	3.2032	2.4702

Table 3 Predicted bugs using entropy measures

Year	Y_o	Y	$YR_{\alpha=0.1}$	$YR_{\alpha=0.3}$	$YR_{\alpha=0.5}$	$YR_{\alpha=0.7}$	$YR_{\alpha=0.9}$	$YT_{\alpha=0.1}$	$YT_{\alpha=0.3}$	$YT_{\alpha=0.5}$	$YT_{\alpha=0.7}$	$YT_{\alpha=0.9}$
2008	45	29	40	38	35	33	30	40	38	35	32	30
2009	27	27	25	25	25	25	26	29	27	26	26	26
2010	41	56	55	55	54	54	55	53	53	53	54	55
2011	53	55	56	56	57	56	56	54	55	55	56	56
2012	80	85	79	80	81	83	84	77	79	81	82	84
2013	68	88	98	96	95	92	89	100	98	95	93	89
2014	129	104	90	93	96	100	102	90	94	97	100	103

6 Assessment of Entropy Measures

The statistical performance and regression coefficients using SPSS for the considered data sets are shown in Table 4.

From Table 4, it is concluded that for Shannon entropy [1], R^2 is maximum, i.e., 0.775, and adjusted R^2 is maximum, i.e., 0.730. For Renyi entropy [2], it is observed that on increasing α from 0.1 to 0.9, the value of R^2 increases from 0.620 to 0.763 and adjusted R^2 also increases from 0.544 to 0.716. For example, $R^2 = 0.775$ implies that 77.5% of the variance in dependent variable is predictable from independent variables, and adjusted $R^2 = 0.730$ implies that there is 73.0% variation in the dependent variable. Similar conclusion can be drawn for Tsallis entropy [3].

Further, we have performed the analysis of variance (ANOVA) test and Tukey test to validate the entropy measure results considered in this paper. The one-way ANOVA is used to only determine whether there are any significant differences between the means of three or more independent (unrelated) groups. To determine which specific groups differed from each other, multiple comparison test, i.e., Tukey's HSD test, is used. It takes into consideration the number of treatment levels, the value of mean square error and the sample size and the statistic q that we look up in a table (studentized range statistic table). Once the HSD is computed, we compare each possible difference between means to the value of the HSD. The difference between two means must equal or exceed the HSD in order to be significant. The formula to compute a Tukey's HSD test is as follows:

Table 4 Statistical performance parameters for entropy measures using simple linear regression

Entropy measures	Parameter (α)	R	R^2	Adjusted R^2	Std. error of the estimate	Regression coefficients	
						A	B
Shannon entropy	–	0.881	0.775	0.730	17.57127	−250.486	112.073
Renyi entropy	0.1	0.787	0.620	0.544	22.86081	−171.528	253.845
	0.3	0.812	0.659	0.590	21.65929	−190.628	280.770
	0.5	0.835	0.697	0.637	20.39309	−209.866	308.490
	0.7	0.860	0.739	0.687	18.93128	−232.461	340.699
	0.9	0.874	0.763	0.716	18.03841	−243.825	361.012
Tsallis entropy	0.1	0.782	0.611	0.534	23.11530	−41.947	15.946
	0.3	0.814	0.663	0.595	21.53316	−72.233	28.378
	0.5	0.841	0.707	0.648	20.06774	−110.451	48.672
	0.7	0.862	0.742	0.691	18.82165	−158.038	80.642
	0.9	0.876	0.767	0.721	17.89233	−216.574	129.226

$$\text{HSD} = q\sqrt{\frac{\text{MSE}}{n}} \tag{5}$$

where MSE: mean square error, n: sample size, and q: critical value of the studentized range distribution

6.1 One-Way ANOVA

We have applied one-way ANOVA for Shannon [1], Renyi [2], and Tsallis entropies [3]. The ANOVA test has been applied to five cases. In case 1, Shannon entropy, Renyi entropy for $\alpha = 0.1$, Tsallis entropy for $\alpha = 0.1$ are considered, and in case 2, Shannon entropy, Renyi entropy for $\alpha = 0.3$, Tsallis entropy for $\alpha = 0.3$ are considered. Other cases are defined similarly taking α as 0.5, 0.7, and 0.9 respectively. In all the cases, Shannon entropy values remain the same as this entropy is independent of α. We set null and alternate hypothesis as

H_o : There is no significant difference between the three means. H_1 : There is a significant difference between the three means. Level of confidence is chosen as 0.05 for ANOVA test. Table 5 depicts the results obtained from ANOVA.

It is observed from Table 5 that the calculated value of F in all the five cases is greater than the critical value of F at 0.05. Thus, the null hypothesis is rejected, and alternate hypothesis is accepted. Thus, there is a significant difference in the means of three different measures of entropy, viz. Shannon's [1], Renyi's [2], and Tsallis's [3] entropies. In order to find out which groups show a significant difference, Tukey's HSD test is applied.

6.2 Tukey's HSD Test

We find 'q' from studentized range statistic table using degree of freedom (within) (ANOVA results) and number of conditions 'c.' Here, in all the cases, degree of freedom (within) is 18, and number of conditions is 3, and sample size 'n' is 7. Thus, 'q' from the table for 0.05 level of confidence is 3.61. HSD is computed using Eq. (5), and MSE (mean square error) values are observed from ANOVA

Table 5 Variability of Shannon, Renyi, and Tsallis entropies using ANOVA

Cases	F(calculated)	p value	F critical
Case 1	61.8925	8.57E-09	3.554557
Case 2	76.91664	1.52E-09	3.554557
Case 3	95.27835	2.66E-10	3.554557
Case 4	121.6313	3.5E-11	3.554557
Case 5	159.5823	3.52E-12	3.554557

Table 6 Results of Tukey test

Cases	MSE	HSD	Shannon entropy (mean)	Renyi entropy (mean)	Tsallis entropy (mean)	Shannon versus Renyi entropy (difference of means)	Renyi versus Tsallis entropy (difference of means)	Tsallis versus Shannon entropy (difference of means)
Case 1	0.945251	1.326574	2.799714	0.925029	6.599174	1.874685	5.674145	3.79946
Case 2	0.34099	0.796762	2.799714	0.904346	4.775417	1.895368	3.871071	1.975703
Case 3	0.140342	0.511154	2.799714	0.885448	3.569534	1.914266	2.684086	0.76982
Case 4	0.069593	0.359948	2.799714	0.868061	2.744516	1.931653	1.876455	0.055198
Case 5	0.043352	0.284094	2.799714	0.850694	2.165657	1.94902	1.314963	0.634057

results. The mean of Shannon entropy is same for all cases as it is independent of α. Using these values, the difference between each pair of means is evaluated. The pairs are defined as follows: Shannon entropy versus Renyi entropy, Renyi entropy versus Tsallis entropy, and Tsallis entropy versus Shannon entropy. Table 6 depicts the values of HSD, MSE, means of the entropies, and difference of means for all the cases.

In case 1, case 2, case 3, and case 5 for all the three pairs, i.e., Shannon versus Renyi, Renyi versus Tsallis, and Tsallis versus Shannon, the HSD value is less than their respective difference of means. Thus, according to Tukey test, the difference between Shannon entropy and Renyi entropy, Renyi entropy and Tsallis entropy, and Tsallis entropy and Shannon entropy are statistically significant. But in case 4, the difference of means of the pair Tsallis versus Shannon is less than the HSD value. Thus, for this case, the difference between Tsallis entropy and Shannon entropy is not statistically significant, whereas the difference between Shannon entropy and Renyi entropy and Renyi entropy and Tsallis entropy are statistically significant.

7 Conclusion and Future Scope

The quality and reliability of the software are highly affected by the bugs lying dormant in the software. After collecting bug reports of each subcomponent from the CVS repository, the number of bugs present in each of them is recorded on yearly basis taken from 2008 to 2014. Using these data sets, the Shannon entropy, Renyi entropy, and Tsallis entropy are calculated. Renyi entropy and Tsallis entropy for five different values of α are considered, i.e., $\alpha = 0.1, 0.3, 0.5, 0.7, 0.9$. It is observed that for this data set obtained, Shannon entropy lies between 2 and 4. Renyi and Tsallis entropies decrease as α value increases. Simple linear regression technique is applied to calculate the predicted bugs using the calculated entropy and observed bugs in SPSS software. These predicted bugs help in maintaining the quality of the software and reducing the testing efforts. We have compared the performance of different measures of entropy considered on the basis of different comparison criteria, namely R^2, adjusted R^2, and standard error of estimate. It is also observed that among Shannon entropy, Renyi entropy, and Tsallis entropy, R^2 is maximum in the case of Shannon entropy, i.e., $R^2 = 0.775$. In the case of Renyi entropy, R^2 is maximum when $\alpha = 0.9$, i.e., $\mathrm{R}^2 = 0.763$. Similarly, for Tsallis entropy also, R^2 is maximum when $\alpha = 0.9$, i.e., $R^2 = 0.767$. By applying ANOVA and Tukey test, the different measures of entropy are validated. In this paper, only open-source Mozilla products are considered. The study may be extended to other open-source and closed-source software systems with different subcomponents. Entropy measures, namely, Shannon entropy, Renyi entropy, and Tsallis entropy, are considered for evaluation of predicted bugs, and the parameter value for Renyi entropy and Tsallis entropy is limited to few values. The study may be extended for other measures of entropy with different parameter values. This study can further be extended to analyze the code change process using other entropy measures in other projects which can be further used in predicting the future bugs in a project.

References

1. Shannon, C.E.: A mathematical theory of communication. Bell Syst. Tech. J. **27**(3), 379–423 (1948). 623–656
2. Renyi, A.: On measures of entropy and information. In: Proceedings 4th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 547–561 (1961)
3. Tsallis, C., Mendes, R, Plastino, A. The role constraints within generalised non extensive statistics. Physica 261A, pp. 534–554 (1998)
4. Goel, A.L., Okumoto, K.: Time dependent error detection rate model for software reliability and other performance measures. IEEE Trans. Reliab. **28**(3), 206–211 (1979)
5. Huang, C.Y., Kuo, S.Y., Chen, J.Y.: Analysis of a software reliability growth model with logistic testing effort function. In: Proceedings of Eighth International Symposium on Software Reliability Engineering, pp. 378–388 (1997)
6. Kapur, P.K., Garg, R.B.: A software reliability growth model for an error removal phenomenon. Softw. Eng. J. **7**, 291–294 (1992)
7. Kapur, P.K., Pham, H., Chanda, U., Kumar, V.: Optimal allocation of testing effort during testing and debugging phases: a control theoretic approach. Int. J. Syst. Sci. **44**(9), 1639–1650 (2013)
8. Kapur, P.K., Chanda, Udayan, Kumar, Vijay: Dynamic allocation of testing effort when testing and debugging are done concurrently communication in dependability and quality management. Int. J. Serbia **13**(3), 14–28 (2010)
9. Hassan, A.E.: Predicting faults based on complexity of code change. In: The proceedings of 31st International Conference On Software Engineering, pp. 78–88 (2009)
10. Ambros, M.D., Robbes, R.: An extensive comparison of bug prediction approaches. In: MSR'10: Proceedings of the 7th International Working Conference on Mining Software Repositories, pp. 31–41 (2010)
11. Singh, V.B., Chaturvedi, K.K.: Bug tracking and reliability assessment system (BTRAS). Int. J. Softw. Eng. Appl. **5**(4), 1–14 (2011)
12. Khatri, S., Chillar, R.S., Singh, V.B.: Improving the testability of object oriented software during testing and debugging process. Int. J. Comput. Appl. **35**(11), 24–35 (2011)
13. Singh, V.B., Chaturvedi, K.K.: Improving the quality of software by quantifying the code change metric and predicting the bugs. In: Murgante, B., et al. (eds.) ICCSA 2013, Part II, LNCS 7972, pp. 408–426. Springer, Berlin (2013)
14. Chaturvedi, K.K., Kapur, P.K., Anand, S., Singh, V.B.: Predicting the complexity of code changes using entropy based measures. Int. J. Syst. Assur. Eng. Manag. **5**(2), 155–164 (2014)
15. Singh, V.B., Chaturvedi, K.K., Khatri, S.K., Kumar, V.: Bug prediction modelling using complexity of code changes. Int. J. Syst. Assur. Eng. Manag. **6**(1), 44–60 (2014)
16. Sharma, M., Kumari, M., Singh, R.K., Singh, V.B.: Multiattribute based machine learning models for severity prediction in cross project context. In: Murgante, B., et al. (eds.) ICCSA 2014, Part V, LNCS 8583, pp. 227–241 (2014)
17. http://bugzilla.mozilla.org
18. Weisberg, S.: Applied linear regression. Wiley, New York (1980)