Method for the Automatic Generation of Software Test Data Based on the Integration of Structural Equation Model

Han Likai[1], and Liu Jun[2]

[1]Xi'an University, 710065

[2]Xi'an University of Posts & Telecommunications, 710061

Keywords: Software Test Data; Automatic Generation; Reliability Evaluation; Structural Equation Model

Abstract: Software reliability evaluation performance directly affects the workload of automatic software test data generation. Therefore, in view of the problem in the automatic generation and correction processing of software test data in the software test data automatic generation work, a kind of method for the automatic generation of software test data based on the integration of the structural equation model is put forward. This method takes the diversity of the software engineering into consideration, makes use of the structural equation model to automatically generate the software test data, and combines the exponential distribution correction time to complete the software test data correction process. Through the test of the two real fault data sets (Ohba and Wood), the proposed method is compared with the existing software reliability growth model (hereinafter referred to as SRGM for short). The results show that the model fitting effect with the integration of the structural equation model is the best, which demonstrates more superior software reliability evaluation performance and model adaptability.

1. Introduction

The automatic generation of software test data has attracted more and more attention in the field of software development at present [1]. As is known to all, software reliability directly affects the software development cost and software quality. The software reliability growth model (hereinafter referred to as SRGM for short) has provided the necessary information for a lot of software development work decision making, such as the cost analysis, the allocation of resources during the test, as well as the time of the decision release [2-3]. The software reliability growth model aims to explain the reaction caused by the failure during the process of the automatic generation of software test data [4-5]. Most of the existing software reliability growth models assume of the automatic generation of the software test data, such as the perfect debug troubleshooting and the direct fault correction, which is inaccurate in the practical work [6]. In the model put forward in this paper, more realistic assumption is set to apply the improved software reliability growth model [7].

In general, different generation patterns for the software test data may be obtained using different non-subtractive averaging functions [8]. The selection of the automatic test pattern generation for the software test data is based on the goodness of fit in the mode to the potential software fault data. SRGM is widely applied to the fault related behaviors of the software system, such as the SRGM for the exponential non-homogeneous poisson process (hereinafter referred to as NHPP for short), the SRGM combined with S type NHPP and so on. However, as the clear majority of the SRGMs are embedded with certain restrictive conditions or assumed conditions, the selection of the appropriate mode in accordance with the characteristics of the software engineering is usually challenging [9]. To choose a suitable model, the following two ways should be taken: First, a guideline is designed so that the fitting model for the software engineering can be put forward; secondly, the mode with the highest confidence level is selected after a few evaluations. If the software engineering is huge and complex, the decision-making process will generate huge costs [10]. To reduce such huge overhead, the method of the structural equation model can be used. The transformation method can be adopted for the structural equations, which can adapt to the fault

  DOI: 10.25236/isrme.2019.032

processing characteristics in accordance with the actual data. The results of the research show that the structural equation model can be used to automatically generate the software test data.

The correction time is an important indicator of the software test data correction process in the automatic generation of the software test data. Most of the SRGMs in the early stage did not take the software test data correction time into consideration or considered that the correction time was a fixed value. Obviously, this is not practical. In this paper, the structural equation model is put forward to complete the automatic generation of the software test data. And the correction process of the software test data is completed in combination with the exponential distribution correction time. The comparison between the method put forward in this paper and the existing SRGMs is carried out through the experiment on two sets of real fault data (Ohba and Wood). The results show that the fitting effect of the model based on the integration of the structural equation model is the most superior, which has demonstrated better software reliability evaluation performance and model adaptability.

2. Method for the Automatic Generation of Software Test Data

The research on the automatic generation of the software test data based on the automatic generation and the correction process of the software test data is carried out. The method put forward in this paper is divided into two phases, that is, the software test data automatic generation process based on the structural equation model and the software test data correction process based on the index correction time.

2.1. Process of the Automatic Generation of Software Test Data

In order to achieve the automatic generation of the software test data that is practical, the following assumptions are made in this paper: 1) In case of the occurrence of fault, the source of the fault that leads to the result that the fault cannot be eliminated immediately; 2) The error correction is not complete. And once the fault is identified, it can be completely repaired at certain probability p for sure; 3) Due to different environmental factors, the automatic generation of the software test data and the use of the stage failure rate is not the same either; 4) The number of times for the detection of the failure in each software is independent.

In the automatic generation mode of the software test data based on the NHPP, the cumulative number of faults $N(t)$ is used to define the non-homogeneous Poisson process (NHPP) through the rate function (also referred to as the intensity function) $\lambda(t)$ and the expected function of the number of the detected faults $m(t) = E(N(t))$, in which $m(t)$ is obtained through the following

$$m(t) = \int_0 \lambda(x)\,dx \tag{1}$$

Structural equations are made up of simple elements of parallel operation. These elements are activated by the biological nervous system. In fact, the structural equation is mainly determined by the association between the elements. We can train the structural equations executes the specific function through the adjustment of the associated values (weights) between elements. In normal situation, we make adjustment to or carry out training on the structural equations to achieve the result that a specific input can lead to a specific target output. The adjustment of the structural equation model is based on the comparison of the output with the target value until the output of the structural equation matches the target value. This is particularly typical in the supervised learning, and many of such types of input/target pairs are used to train the structural equations. In the applications of various fields, the structural equations are trained to perform the complex value function, including the pattern recognition, the identification, the classification, the view and the control system.

The structural equations fall under the category of the learning mechanism, which can approach any nonlinear continuous function on the basis of the known data. Generally speaking, the structural equation is composed of the following three main components:

Structural equations: Each structural equation can receive a signal, carry out the processing, and finally generate an output signal. Figure 1 depicts a structural equation, in which f stands for the activation function for the processing of the input signal and the generation of the output; x_i stands for the output of the structural equation model in the upper layer; and w_i stands for the associated weight of the structural equation in the upper layer.

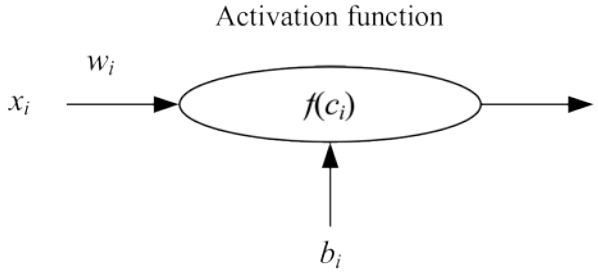

Figure 1 Diagram on the structural equation model

The architecture of the structural equation system is as the following: The most common type of the structural equation system architecture is the feedforward structural equation, as shown in Figure 2. This kind of system architecture is composed of three significant layers: an input layer, a hidden layer and an output layer. It is worth noting that the circle stands for the structural equations, and the connection of the cross-layer structural equation model is referred to as the associated weight.

Learning algorithm: This algorithm describes the process of the adjustment to the weight. In the learning process, the weight of the structural equation is adjusted to reduce the structural equation output error compared to the standard answer. The back propagation algorithm is one of the most widely used. In the algorithm, the structural equation weight is repeatedly trained in accordance with the back propagation error of the output layer.

The goal of the application of the structural equation model is to roughly estimate the nonlinear function that can receive the vector $X = (x_1, x_2, K, x_n)$ and the output vector $Y = (y_1, y_2, K, x_m)$.

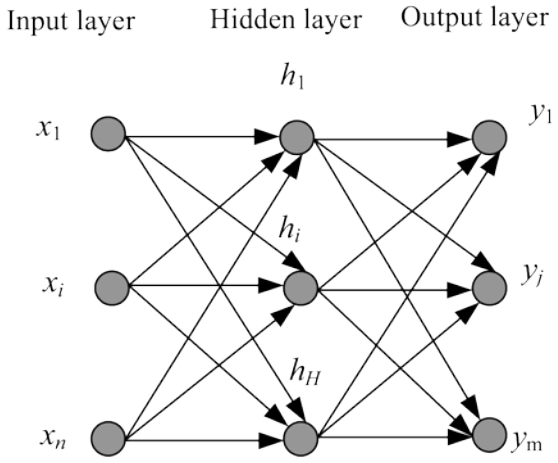

Figure 2 Structure of the feedforward structural equation

$$c_i = b + \sum_{j=1}^{n} x_j w_j \tag{2}$$

Hence, we define the structural equation as $Y = F(X)$. And the composition of Y can be obtained by the following equation:

$$y_k = g\left(b_k + \sum_{j=1}^{h} w_{jk}^0 h_j\right), k = 1,\ldots m, \tag{3}$$

In the equation, w_{jk}^{0} stands for the output weight from the hidden layer node j to the output layer node k ; h_j stands for the output of the hidden layer j ; b_k stands for the deviation rate of the output node k ; g stands for the activation function in the output layer. And the value in the hidden layer is obtained by the following equation:

$$h_j = f\left(b_j + \sum_{i=1}^{n} w_{ij}^{1} x_i\right), j = 1,2,\ldots,H, \tag{4}$$

In the equation: w_{ij}^{1} stands for the input weight from the input layer node i to the hidden layer node j , x_i stands for the value of i in the input node, b_j stands for the deviation rate of the node j in the hidden layer, and f stands for the activation function in the hidden layer.

As the structural equation approximation function can be used as a nested function $f\left(g(x)\right)$, it can be applied to the software reliability model. Since the latter is aimed to establish a kind of model that can explain the behavior of the software faults. In other words, if we obtain a compound function form from the ordinary SRGM, we can construct a model on the basis of the structural equation model for the software reliability. In this paper, the structural equation model is used. And the mean function of this model is obtained by the equation as the following:

$$m(t) = \frac{a}{1 + ke^{-bt}} \tag{5}$$

In the above equation, a, b and k are all positive real numbers.

By replacing with k with e^{-c} , we can easily obtain the following compound function from the mean function:

$$m(t) = \frac{a}{1 + ke^{-bt}} = \frac{a}{1 + e^{-(bt+c)}} \tag{6}$$

It is worth pointing out that the mean function $m\ (t)$ is composed of the following functions

$$g(x) = b(x) + c \tag{7}$$

$$f(x) = \frac{a}{1 + e^{-x}} \tag{8}$$

$$k(x) = a(x) \tag{9}$$

Hence, the following can be obtained

$$m(t) = k\left(f\left(g(t)\right)\right) = \frac{a}{1 + e^{-(bt+c)}} \tag{10}$$

At this point, through the basic feedforward structural equation (as shown in Figure 2), the compound function is obtained from the perspective of the structural equations. However, as shown in Figure 3, the structural equation has only one structural equation in each layer. The input and the output of the hidden layer can be obtained by two equations as the following:

$$h_{in}(t) = w_{11}^{1} t + b_1 \tag{11}$$

$$h(t) = f\left(h_{in}(t)\right) \tag{12}$$

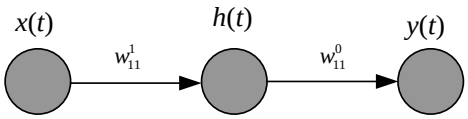

Figure 3 Feedforward structural equation with the single structural equation model in each layer

The equation $f(x)$ stands for the activation function in the hidden layer. Then the input and output of the output layer are as the following

$$y_{in}(t) = w_{11}^{0} h(t) + b_0 \tag{13}$$

$$y(t) = g\left(y_m(t)\right) \tag{14}$$

If we set the activation function as the following

$$f(x) = \frac{1}{1+e^{-x}} \tag{15}$$

$$g(x) = x \tag{16}$$

The deviation rate in the output layer is removed. And then the following result can be obtained:

$$y(t) = y_{in}(t) = \frac{w_{11}^{0}}{1+e^{-\left(w_{11}^{1} t + b_1\right)}} \tag{17}$$

Therefore, we can establish the structural equation model from the logistic growth curve.

The fault prediction steps on the basis of the structural equation model are as the following: 1) Through the definition of the deviation rate and the ubiquitous activation function, the structural equation model is constructed from the logistic growth curve and the model parameters are estimated; 2) Through the time interval data of the system and the cumulative number of faults, the back propagation algorithm is used to train the structural equations; 3) After the training of the structural equations, the test time feedback is sent to the structural equations, and the input is the number of faults predicted within the given time after the training of the structural equations.

2.2. Software Test Data Correction Process

Since the fault can only be excluded after the inspection, it is more appropriate to assume that the software test data calibration process is associated with the automatic software test data generation process. It is assumed that the software test data calibration process is the time delay software test data generation process. In the past, there have been many studies that put forward different models in different forms for the different time delays in these processes. In the literature[8], the software test data are generated automatically through the non-homogeneous Poisson process (NHPP) modeling. And there is constant time lag between the software test data correction and the automatic generation time of the software test data. We use $\Delta(t)$ to stand for the delay time. This kind of delay can be modeled as either a deterministic or random variable, or it can be time dependent. The correction process for the software test data using the mean value function $m_c(t)$ modeling can be obtained from $m(t)$ and $\Delta(t)$.It is assumed that every fault detected requires the same time to correct, that is to say, $\Delta(t) = \triangle$. Therefore, given the automatic generation rate $\lambda(t)$ of the software test data, the strength function of the software test data correction can be obtained in accordance with the following equation:

$$\lambda_c(t) = \begin{cases} \lambda(t) & t < \triangle \\ \lambda(t-\triangle) & t \geq \triangle \end{cases} \tag{18}$$

Therefore, the mean function of the fault correction process can be obtained in accordance with the following equation:

$$m_c(t) = m(t-\triangle), t \geq \triangle \tag{19}$$

The time lag between the automatic generation of the software test data and the correction of the software test data can be related to time. When the more difficult the detected fault can be corrected, the longer the correction time will become longer. In this case, we assume that the time lag is the equation as the following:

$$\Delta(t)=\frac{log(1+\alpha t-\alpha/\beta)}{\beta} \quad \alpha\langle\beta \tag{20}$$

In the equation: α and β stand for the parameters that need to be estimated. Therefore, the correction rate and the mean function can be obtained as the following

$$\lambda_c(t)=\lambda(t-\triangle t) \tag{21}$$

$$m_c(t)=\int_{\triangle}^{t}\lambda(x-\triangle(x))dx \tag{22}$$

In normal situation, the deterministic correction time in practice is not realistic. And the software correction time is closely related to the uncertain factor of human beings. In addition, the faults detected during the system testing are not the same, and the display sequence is random. Therefore, it is more realistic to correct the time with the random variable modeling.

As is known to all, the correction time probably follows an exponential distribution. Therefore, we assume that the correction time of each detected software test data is a random variable $\Delta(t)exp(\mu)$ that follows the exponential distribution. Hence, the correction rate and the mean function can be obtained as the following respectively

$$\lambda_c(t)=E\left[\lambda(t-\triangle t)\right]=\int_{0}^{\infty}\lambda(t-x)\mu e^{-\mu x}dx \tag{23}$$

$$m_c(t)=\int_{\triangle}^{t}\lambda_c(x)dx \tag{24}$$

3. Case Validation

In this section, we make use of two sets of real fault data sets, that is, the Ohba set and the Wood set, to carry out the statistical analysis and the model fitting analysis on the method put forward in this paper. The two sets of real data sets contain three types of data, that is, the test time, the test workload, as well as the number of detection faults, which is the classical instance of the software reliability model performance comparison test. Data record up to 20 weeks have been selected for both the fault data sets to carry out the reliability with the two kinds of NHPP SRGMs that are commonly used at present.

In order to verify the automatic software test data generation and the correction model put forward in this paper, we first need to apply the maximum likelihood estimation method to estimate the parameters of the model. In the experiment of this paper, the structural equation derived from the structural equation model is applied. In the mean value function of the structural equation model, the maximum likelihood estimation is used to estimate the parameters a , b and c of the equation (10), which are $\hat{a}$ 、$\hat{b}$ and $\hat{c}$, respectively. Similarly, the maximum likelihood estimation method is applied to the exponential distribution parameter of the software test data correction model (*t*) for the estimation. In Table 1, the estimated parameters of the fault and the correction model are set out. From the table, it can be seen that the structural equation software test data automatic generation model and the constant correction time software test data correction model have jointly provided the parameters that are most suitable for the data set in this paper.

Table 1 Estimated Values of the Model Parameters

Model Parameter	Maximum Likelihood Estimation
Distribution of the index	$\hat{a}$ =1467.32
	$\hat{b}$ =0.89
	$\hat{c}$ =31.87
$\Delta(t)\sim exp(\mu)$	$\hat{\mu}$ =0.12

Three kinds of commonly used evaluation criteria are selected to evaluate the results of the model fitting, which are as the following: the squared sum of the mean error, as shown in the equation (18); the correlation index of the regression curve equation, as shown in the equation (19); and the relative error diagram, as shown in the equation (20).

$$M_{SE} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - y_i^{'}\right)^2 \tag{25}$$

In the equation: y_i stands for the actual observed data, y_i' stands for the fitting value estimated by the same model. The smaller the numerical value of M_{SE} is, the better the fitting performance of the model is, that is, the better the reliability evaluation performance of the model is.

$$R_{Sq} = 1 - \frac{\sum_{i=1}^{n}\left(y_i - y_i^{'}\right)^2}{\sum_{i=1}^{n}\left(y_i - y_{ave}\right)^2} \tag{26}$$

In the equation: y_{ave} stands for the mean value of y i, the closer the value of R ^{Sq} to 1 is, the better the reliability evaluation performance is.

$$R_E(i) = \frac{y^{'} - y_i}{y_i} \tag{27}$$

The closer the R_E curve is to the x axis, the better the reliability evaluation performance is.

Table 2 sets out the model fitting results for each model on the two sets of fault data sets.

Table 2 Comparison of the Model Fitting Results for Different Models

Model	Ohba		Wood	
	M_{SE}	R_{Sq}	M_{SE}	R_{Sq}
Logistic SRGM	114.03	0.989	21.45	0.978
Exponential Weibull SRGM	112.31	0.990	13.22	0.985
The method put forward in this paper	98.86	0.993	8.92	0.990

From the fitting results on the Ohba dataset, it can be seen that compared to the other two SRGMs in the experiment, the fitting result of the method put forward in this paper is the best. The minimum value of M_{SE} is 98.86, and the value of R_{Sq} that is closest to 1 is 0.993. Figure 4 shows the comparison of the fitted values of the mean number of failures detected in these two models and the method put forward in this paper for the Ohba dataset. It can be seen that the overall fit between the fitting curve of the method put forward in this paper and the actual observed value curve is relatively high. Figure 5 shows the comparison of the R_E curves of the three methods on each data point in the Ohba dataset. It is clear that the R_E curve of the method put forward in this paper is closest to the *x* axis.

From the fitting results on the Wood dataset, it can be seen that compared to the other two kinds of SRGMs in the experiment, the fitting effect of the method put forward in this paper is the best. The minimum value of M $_{SE}$ is 8.92, and the value closest to 1 is 0.990. Figure 6 shows the comparison of the fitting values of the mean number of the defects detected in these two models and the method put forward in this paper for the Wood dataset. It can be seen that the fitting curve of the method put forward in this paper has relatively high overall fit to the curve of the actual observed value. Figure 7 shows the comparison of the R_E curves of the three methods on each data point in the Wood dataset. It is clear that the R_E curve of the method put forward in this paper is closest to the x axis.

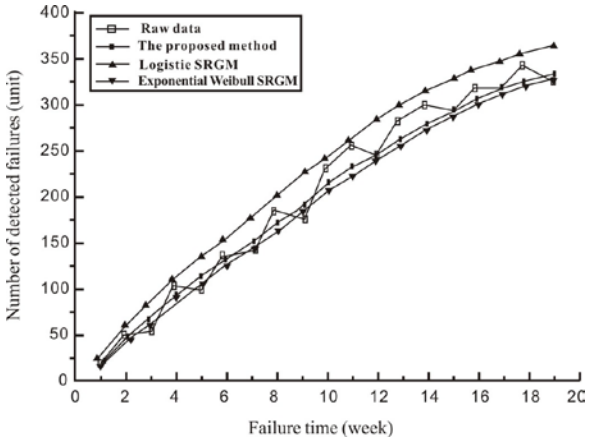

Figure 4 Fitting Result of the Cumulative Number of Failures on the Ohba Data Set

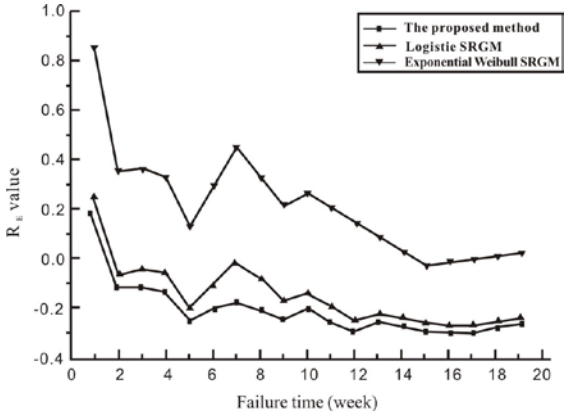

Figure 5 Results of R_E on the Ohba Dataset

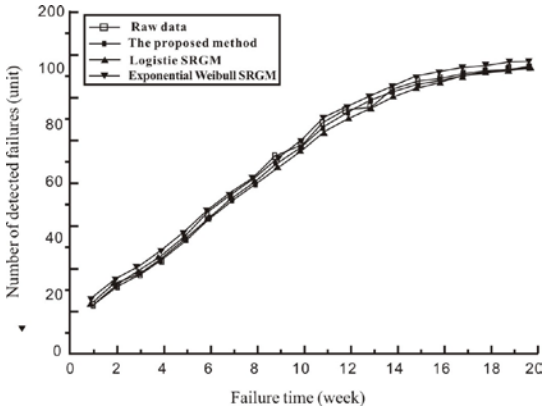

Figure 6 Fitting Result of the Cumulative Number of Failures on the Wood Data Set

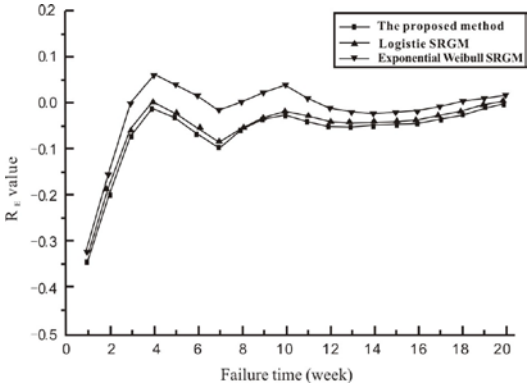

Figure 7 Results of R_E on the Wood Dataset

The software reliability test method based on the structural equation model put forward in this paper has the optimal fitting effect on the two data sets, which is significantly superior to that of the comparison models. Therefore, it is demonstrated that the method put forward in this paper has very good effectiveness and applicability in the model evaluation performance.

4. Conclusions

In this paper, the problem of automatic generation of software test data is studied, and a kind of method for the automatic generation of software test data based on the integration of structural equation model is put forward. This method takes the diversity of the software engineering into consideration, makes use of the structural equation model to automatically generate software test data, and combines the exponential distribution correction time to complete the software test data correction process. The test of the actual software fault data shows that the performance based on the integration of the structural equation model is superior to that of the existing software reliability growth models.

References

[1] Focaccia, S., Tinti, F., & Bruno, R. (2013). A software tool for geostatistical analysis of thermal response test data: ga-trt. Computers & Geosciences, 59(3), 163-170.

[2] Mann, M., Tomar, P., & Sangwan, O. P. (2017). Bio-inspired metaheuristics: evolving and prioritizing software test data. Applied Intelligence, 1-16.

[3] TThomson, B. J., & Lang, N. P. (2016). Volcanic edifice alignment detection software in matlab: test data and preliminary results for shield fields on venus. Computers & Geosciences, 93, 1-11.

[4] Marsden, W. (2015). A software architecture for managing the material information data streams, test data and model predictions, evident in successful icme implementations. Annals of Surgery, 55(3), 421-433.

[5] Smíšek, R., Maršánová, L., Němcová, A., Vítek, M., Kozumplík, J., & Nováková, M. (2016). Cse database: extended annotations and new recommendations for ecg software testing. Medical & Biological Engineering & Computing, 55(8), 1-10.

[6] Bueno, P. M. S., Jino, M., & Wong, W. E. (2014). Diversity oriented test data generation using metaheuristic search techniques. Information Sciences, 259(3), 490-509.

[7] Lv, J., Yin, B. B., & Cai, K. Y. (2014). On the asymptotic behavior of adaptive testing strategy for software reliability assessment. IEEE Transactions on Software Engineering, 40(4), 396-412.

[8] Pan, K., Wu, X., & Xie, T. (2014). Guided test generation for database applications via synthesized database interactions. ACM Transactions on Software Engineering and Methodology (TOSEM), 23(2), 1-27.

[9] Kok, G. J. P., Harris, P. M., Smith, I. M., & Forbes, A. B. (2016). Reference data sets for testing metrology software. Metrologia, 53(4), 1091-1100.

[10] Zhou, Z. Q., Xiang, S., & Chen, T. (2016). Metamorphic testing for software quality assessment: a study of search engines. IEEE Transactions on Software Engineering, 42(3), 264-284.