Scientific Research Publishing

Journal of Software Engineering and Applications, 2021, 14, 11-25
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

OSS Effort Expense Optimization Based on Wiener Process Model and GA

Kodai Sugisaki[1], Yoshinobu Tamura[1], Shigeru Yamada[2]

[1]Tokyo City University, Tokyo, Japan
[2]Tottori University, Tottori, Japan
Email: g1981826@tcu.ac.jp, tamuray@tcu.ac.jp, yamada@tottori-u.ac.jp

How to cite this paper: Sugisaki, K., Tamura, Y. and Yamada, S. (2021) OSS Effort Expense Optimization Based on Wiener Process Model and GA. *Journal of Software Engineering and Applications*, **14**, 11-25.
https://doi.org/10.4236/jsea.2021.141002

Received: December 7, 2020
Accepted: January 15, 2021
Published: January 18, 2021

Abstract

Various open source software are managed by using several bug tracking systems. In particular, the open source software extends to the cloud service and edge computing. Recently, OSF Edge Computing Group is launched by OpenStack. There are big data behind the internet services such as cloud and edge computing. Then, it is important to consider the impact of big data in order to assess the reliability of open source software. Various optimal software release problems have been proposed by specific researchers. In the typical optimal software release problems, the cost parameters are defined as the known parameter. However, it is difficult to decide the cost parameter because of the uncertainty. The purpose of our research is to estimate the effort parameters included in our models. In this paper, we propose an estimation method of effort parameter by using the genetic algorithm. Then, we show the estimation method in section 3. Moreover, we analyze actual data to show numerical examples for the estimation method of effort parameter. As the research results, we found that the OSS managers would be able to comprehend the human resources required before the OSS project in advance by using our method.

Keywords

Fault Big Data, Cost Optimization, Reliability Analysis, Wiener Process Model, Open Source Project

1. Introduction

Recently, the network-oriented open source software (OSS) has been used many users. In particular, the cloud and edge computings attract the extensive attention of the developers and users, because of the network revolution such as 5 G.

Also, the OSS is helpful for many users to make a cost reduction, standardization, and quick delivery. In particular, several research papers in terms of cloud and edge computings such as "OpenStack" are proposed in the past [1] [2] [3]. From past to present, the development and maintenance paradigm of OSS has been changed with the times.

Historically, the software reliability growth models have been applied to the system testing phase of software development [4]. Also, our research team has developed several reliability assessment methods [5] [6] [7]. Moreover, several optimal software release problems have been discussed by the specified researchers [8]. Furthermore, our research team has discussed several optimum release problems. However, the cost parameters have been assumed as the known parameters considering the optimum software release problem. Then, we propose an estimation method of cost parameters based on the genetic algorithm. Thereby, we can obtain the software effort expense required for the OSS management. Considering the characteristics of OSS such as the cost reduction and standardization, the OSS will increasingly give an advantage in the cloud and edge computing. Then, there is a need to consider the influences from the big data behind the OSS management.

As the research background, it is difficult for the OSS managers to comprehend the human resources required before the OSS project in advance. Especially, the environment of software development in the cloud and edge computing will be the complex situation. Also, there are several optimal release problems based on the software reliability growth models. Then, the cost parameters are given in the past researches. On the other hand, we can appropriately control the human resources, if the software managers can estimate the effort parameters in advance.

The problems and solutions in order to solve in the paper are listed as follows:

1) the estimation method of the given parameters included in the existing optimal release problems;

2) the optimal release problem with application to the effort expense optimization;

3) the optimal solution by using GA.

This paper discusses a method of OSS project management considering irregular fluctuation via the big data arose from OSS development and management under the cloud and edge computing. In particular, the effort expense optimization method based on Wiener process model in terms of effort is proposed in order to estimate the effort parameters in this paper. Then, the genetic algorithm is applied to the proposed method. Also, we analyze actual software effort expense data by showing numerical examples of OSS project optimization analysis.

2. Wiener Process Model Description

In the assumptions of Wiener process modeling, we consider the following situations:

1) the noisy event of software effort expense arise from the complex management in big data environment;

2) the factor of big data is defined as "3 V model" such as Volume, Variety, and Velocity;

3) the velocity directly influence the software effort, the variety and volume indirectly influence one.

Moreover, it is important to consider the effort control, because several research papers considering the relationship between effort and fault have been proposed by in the past [9] [10] [11]. Also, the Wiener process has been useful in precision required high accuracy. As the other examples of applying the Wiener process effectively, several research papers have been proposed in the past [12].

Then, we can obtain the following two dimensional stochastic differential equation with Brownian motion derived from software reliability growth modeling approach based on Wiener process [13]:

$$\begin{aligned}\mathrm{d}\Psi(t) &= \left\{\zeta(t) - \frac{1}{2}\left(\upsilon_1^2 + \upsilon_2^2\right)\right\}\left\{\lambda - \Psi(t)\right\}\mathrm{d}t \\ &\quad + \upsilon_1\left\{\lambda - \Psi(t)\right\}\mathrm{d}\omega_1(t) + \upsilon_2\left\{\lambda - \Psi(t)\right\}\mathrm{d}\omega_2(t).\end{aligned} \tag{1}$$

Then, each parameter is as follows:

$\Psi(t)$: the cumulative maintenance effort expense at up to operational time; $t(t \geq 0)$ in the OSS development project, which takes on continuous real values;

$\zeta(t)$: the increase rate of maintenance effort expense at operational time t and a non-negative function;

λ: the estimated maintenance effort required until the end of operation;

$\omega_1(t)$: 1st Wiener process considering the "Volume" factor;

$\omega_2(t)$: 2nd Wiener process considering the "Variety" factor;

υ_1: 1st positive constant for "Volume" representing a magnitude of the irregular fluctuation;

υ_2: 2nd positive constant for "Variety" representing a magnitude of the irregular fluctuation.

Moreover, we define the increase rate of maintenance effort expense in case of $\zeta(t)$. We assume the following equations based on software reliability models [4] as the reliability growth function of the proposed model:

$$\zeta(t) \doteq \frac{\frac{\mathrm{d}I(t)}{\mathrm{d}t}}{o - I(t)} = \frac{p}{1 + q \cdot \exp(-pt)}. \tag{2}$$

Then, the parameters in Equation (2) are given by

$I(t)$: the mean value function of the inflection S-shaped software reliability growth model based on a nonhomogeneous Poisson process (NHPP);

o: the expected number of latent faults;

p: the fault detection rate per fault;

q: is defined as $\frac{1-r}{r}$;

r: the impact parameter in terms of "Velocity".

Therefore, the cumulative maintenance effort expense up to time t is obtained as follows:

$$\Psi(t)=\lambda\left[1-\frac{1+p}{1+q\cdot\exp(-pt)}\cdot\exp\left\{-pt-\upsilon_1\omega_1(t)-\upsilon_2\omega_2(t)\right\}\right]. \tag{3}$$

Similarly, the estimated maintenance effort expense required until the end of operation can give as follows:

$$\Psi_r(t)=\lambda\left[\frac{1+p}{1+q\cdot\exp(-pt)}\cdot\exp\left\{-pt-\upsilon_1\omega_1(t)-\upsilon_2\omega_2(t)\right\}\right]. \tag{4}$$

Above mentioned model parameters can easily estimate by using the method of maximum likelihood [12].

3. Effort Expense Optimization Based on Genetic Algorithm

We discuss the effort expense optimization based on the conventional optimal software release problems. Generally, it is interesting for the software developers to estimate the time when we should stop the testing phase in order to release software efficiently. Therefore, several researchers have discussed about the determination of software release times minimizing the expected total software cost [8].

According to the conventional optimal release problems, we can consider the optimal software maintenance problem with software effort expense from the relationship between cost and effort. Our research group has proposed several optimal maintenance problem based on software effort expense. Then, we define the following effort parameters:

τ_1: the fixing effort expense per fault during the operation;

τ_2: the effort expense per unit time during the operation;

τ_3: the maintenance effort expense per fault after the maintenance. Then, the software effort in the operation can be formulated as:

$$\Theta_1(t)=\tau_1\Psi(t)+\tau_2 t. \tag{5}$$

Also, the software maintenance effort expense after the maintenance is represented as follows:

$$\Theta_2(t)=\tau_3\Psi_r(t). \tag{6}$$

Consequently, from Equations (5) and (6), the total software maintenance effort expense is given by

$$\Theta(t)=\Theta_1(t)+\Theta_2(t). \tag{7}$$

The optimum maintenance time t^* is obtained by minimizing $\Theta(t)$ in Equation (7).

In the past, the effort expense parameter has been assumed as the given parameter according to the conventional optimal software release problem. We propose the estimation method of the effort parameters based on genetic algorithm.

Then, we consider as the search problem minimizing the total software maintenance effort expense in Equation (7).

In the past, we have proposed several parameter estimation methods for jump diffusion process models [12]. Then, we focus on a genetic algorithm for the estimation of the effort expense parameters included in the total software maintenance effort expense in Equation (7). The procedure of parameter estimation based on genetic algorithm is given in the following [14]:

Step 1. The initial values are randomly generated;

Step 2. The crossover is executed;

Step 3. The value of goodness-of-fit is calculated from the following evaluation function κ_i included parameters:

$$\begin{aligned} &\min_{\delta} \kappa_i(\delta), \\ &\kappa_i = \sum_{i=0}^{I} \left\{\Theta(i) - \alpha_i\right\}^2 \quad (i = 1, 2, \cdots, I). \end{aligned} \tag{8}$$

Then, the parameters included in evaluation function are as follows:

$\Theta(i)$: the cumulative amount of maintenance effort expense at up to operation time i in Equation (4);

α_i: i-th actual cumulative amount of maintenance effort expense;

δ: the set of effort expense parameters τ_1, τ_2, and τ_3.

Step 4 and Step 2-Step 3 are continued until minimizing the value of evaluation function κ_i.

Then, the fitness function based on our model is formulated as follows:

$$\begin{cases} \min\limits_{\tau_1,\tau_2,\tau_3} \kappa_i(\tau_1, \tau_2, \tau_3), \\ \kappa_i(\tau_1, \tau_2, \tau_3) = \sum\limits_{i=0}^{I} \left\{\Theta(i) - \alpha_i\right\}^2. \end{cases} \tag{9}$$

Specifically, $\kappa_i(\tau_1, \tau_2, \tau_3)$ in Equation (9) is given by the following equation

$$\begin{aligned} &\kappa_i(\tau_1, \tau_2, \tau_3) \\ &= \sum_{i=0}^{I} \left\{\Theta(i) - \alpha_i\right\}^2 = \sum_{i=0}^{I} \left\{\Theta_1(i) + \Theta_2(i) - \alpha_i\right\}^2 \\ &= \sum_{i=0}^{I} \left\{\tau_1 \Psi(i) + \tau_2 i + \tau_3 \Psi_r(i) - \alpha_i\right\}^2 \\ &= \sum_{i=0}^{I} \left\{\tau_1 \lambda \left[1 - \frac{1+p}{1+q\cdot\exp(-pi)} \cdot \exp\left\{-pi - \upsilon_1\omega_1(i) - \upsilon_2\omega_2(i)\right\}\right] + \tau_2 i \right. \\ &\quad \left. + \tau_3 \lambda \left[\frac{1+p}{1+q\cdot\exp(-pi)} \cdot \exp\left\{-pi - \upsilon_1\omega_1(i) - \upsilon_2\omega_2(i)\right\}\right] - \alpha_i\right\}^2. \end{aligned} \tag{10}$$

Then, the effort expense parameters τ_1, τ_2, and τ_3 are searched by the genetic algorithm under the flow in **Figure 1**.

4. Numerical Examples for Effort Expense Optimization

We show several numerical examples by using the effort expense data of Apache

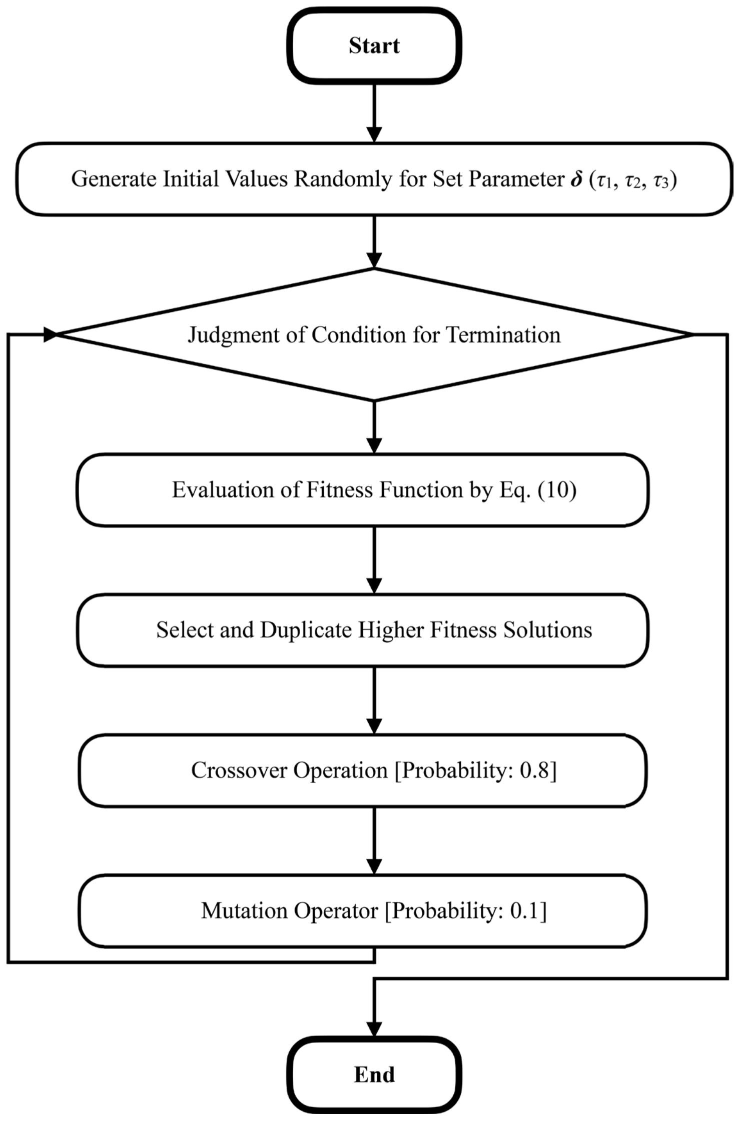
Start
Generate Initial Values Randomly for Set Parameter $\boldsymbol{\delta}$ (τ_1, τ_2, τ_3)
Judgment of Condition for Termination
Evaluation of Fitness Function by Eq. (10)
Select and Duplicate Higher Fitness Solutions
Crossover Operation [Probability: 0.8]
Mutation Operator [Probability: 0.1]
End

Figure 1. Flow of effort expense parameters estimation in Equation (7) based on genetic algorithm.

HTTP Server Project [15]. The actual effort data set is used in the proposed method. The effort data sets are obtained from the bug tracking system of actual OSS project. The effort data requires the data in terms of the person and time. Then, the data sets in terms of the person and time are obtained from "Opened", "Reporter", and "Assignee" recorded on the bug tracking system. We have arranged the effort data from fault big data recorded on the bug tracking system. Then, the unit of effort is ("man × days"), because the unit of date is the day. First, we confirm the state of changing for several effort parameters. Figures 2-4 show the sensitivity analysis for effort expense parameter τ_1, τ_2, and τ_3, respectively. We find that the effort expense parameters τ_1 and τ_2 affect the second half of the operation from Figure 2 & Figure 3. On the other hand, the effort expense parameter τ_1 affect the first half of the operation from Figure 4. In particular, the optimum maintenance times strongly depend on the τ_1, τ_2,

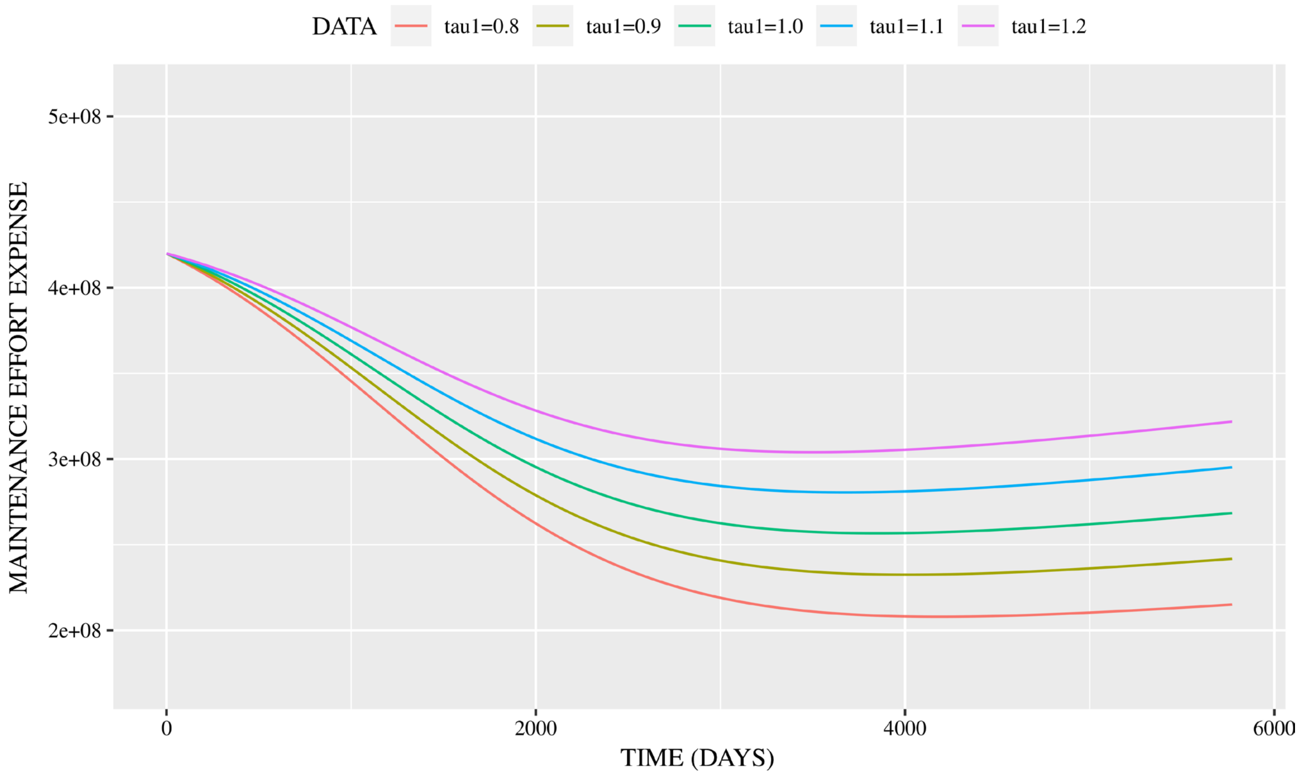

Figure 2. The sensitivity analysis for effort expense parameter τ_1.

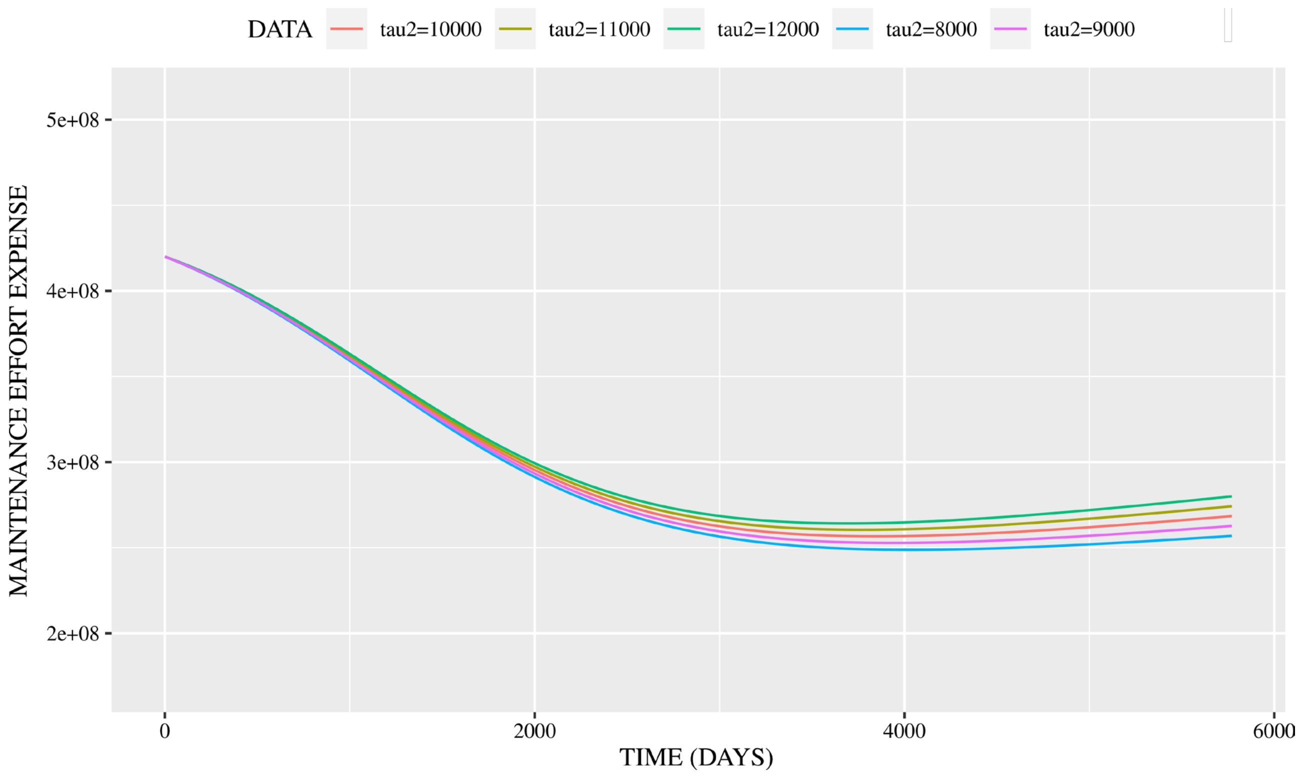

Figure 3. The sensitivity analysis for effort expense parameter τ_2.

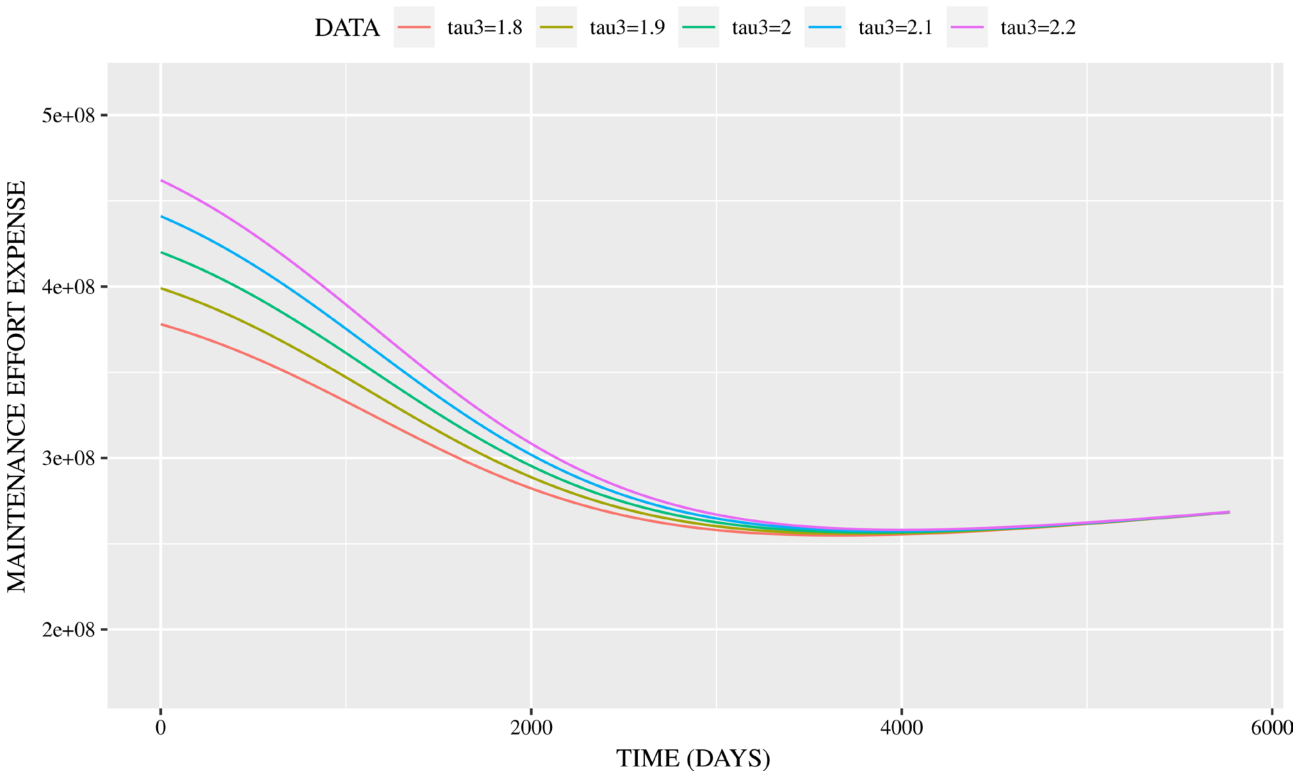

Figure 4. The sensitivity analysis for effort expense parameter τ_3.

and τ_3, respectively. Similarly, we show the sensitivity analysis of the sample paths for effort expense parameter τ_1, τ_2, and τ_3 in **Figures 5-7**, respectively. From **Figures 5-7**, we find that the noises become large in the early phase of operation.

In the past, the effort or cost parameters are the given ones. Conversely, we consider that the effort parameters can be estimated by using Equation (9). **Figure 8** and **Figure 9** show the estimated software effort based on genetic algorithm in Equations (5) and (6). Moreover, we show the estimated total software effort based on genetic algorithm in **Figure 10**. Furthermore, we show the convergence status in **Figure 11**. Then, the effort parameters are estimated by using Equations (8)-(10) in section 3 as follows:

$$\tau_1 = 0.90915, \tau_2 = 10130.2, \tau_3 = 2.19561.$$

From the estimated results, we found that the effort parameters are appropriately estimated by using the combination optimization problem. Then, the software manager will be able to allocate the human resources for the software development appropriately. In other words, the OSS managers can comprehend the maintenance effort expense required in advance by using the proposed method. Therefore, the OSS managers will be able to prepare the human resources required before the OSS project.

The key significance points of the research contribution is as follows:

- There is no research paper in terms of the estimation method of cost parameters and effort ones based on the software reliability growth models. The

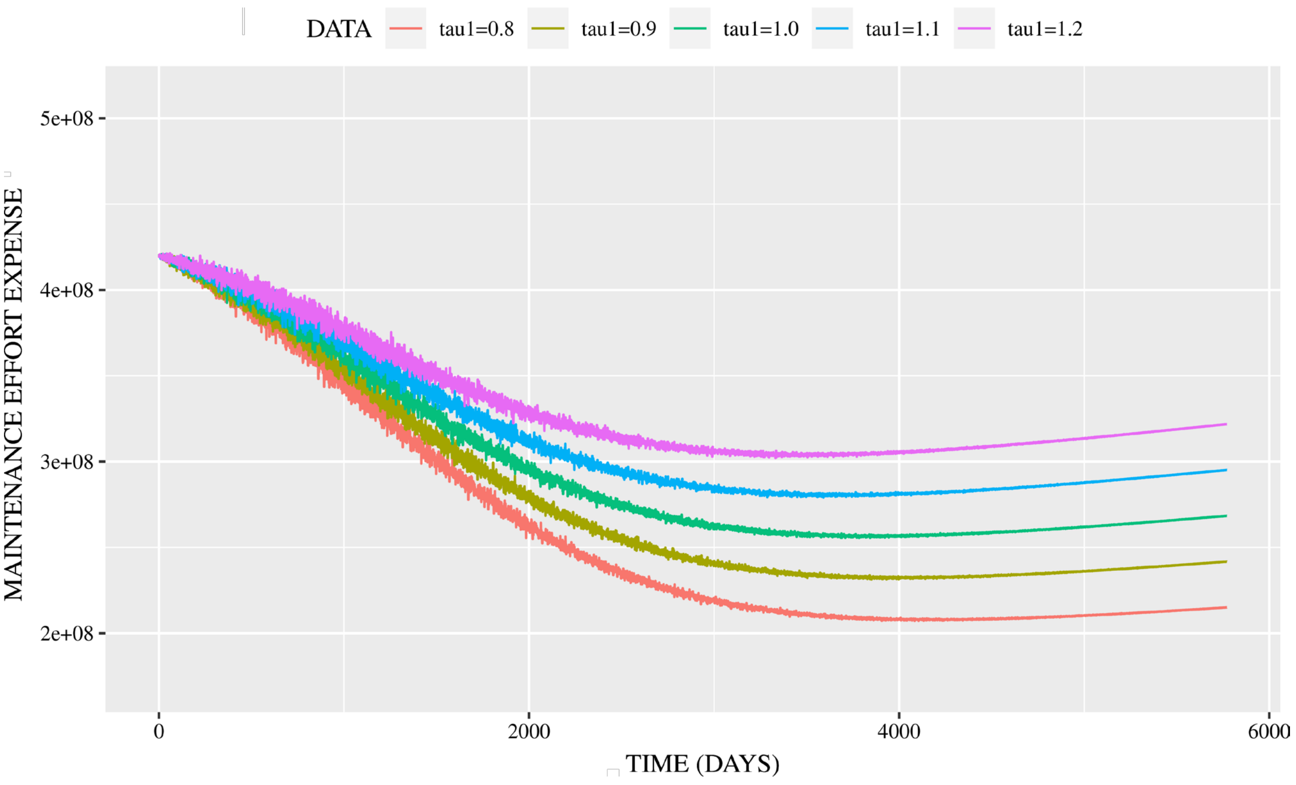

Figure 5. The sensitivity analysis of the sample paths for effort expense parameter τ_1.

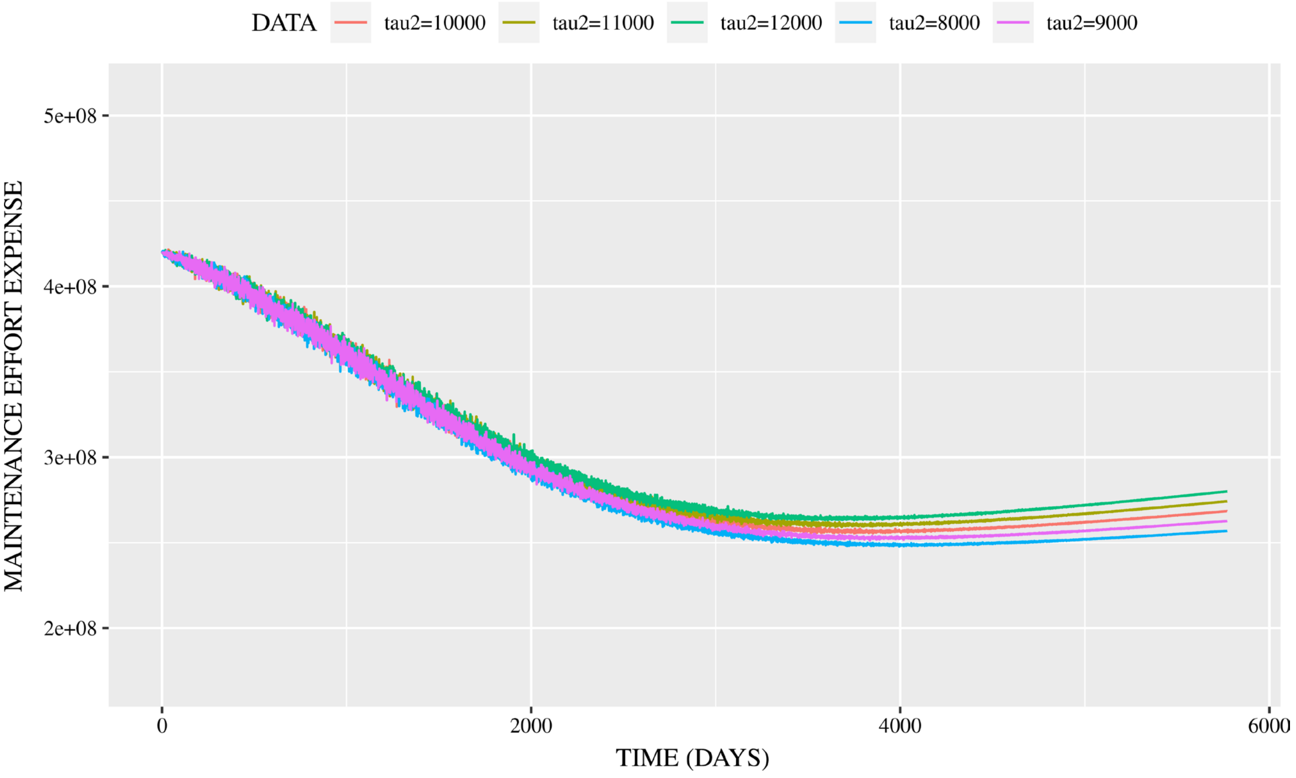

Figure 6. The sensitivity analysis of the sample paths for effort expense parameter τ_2.

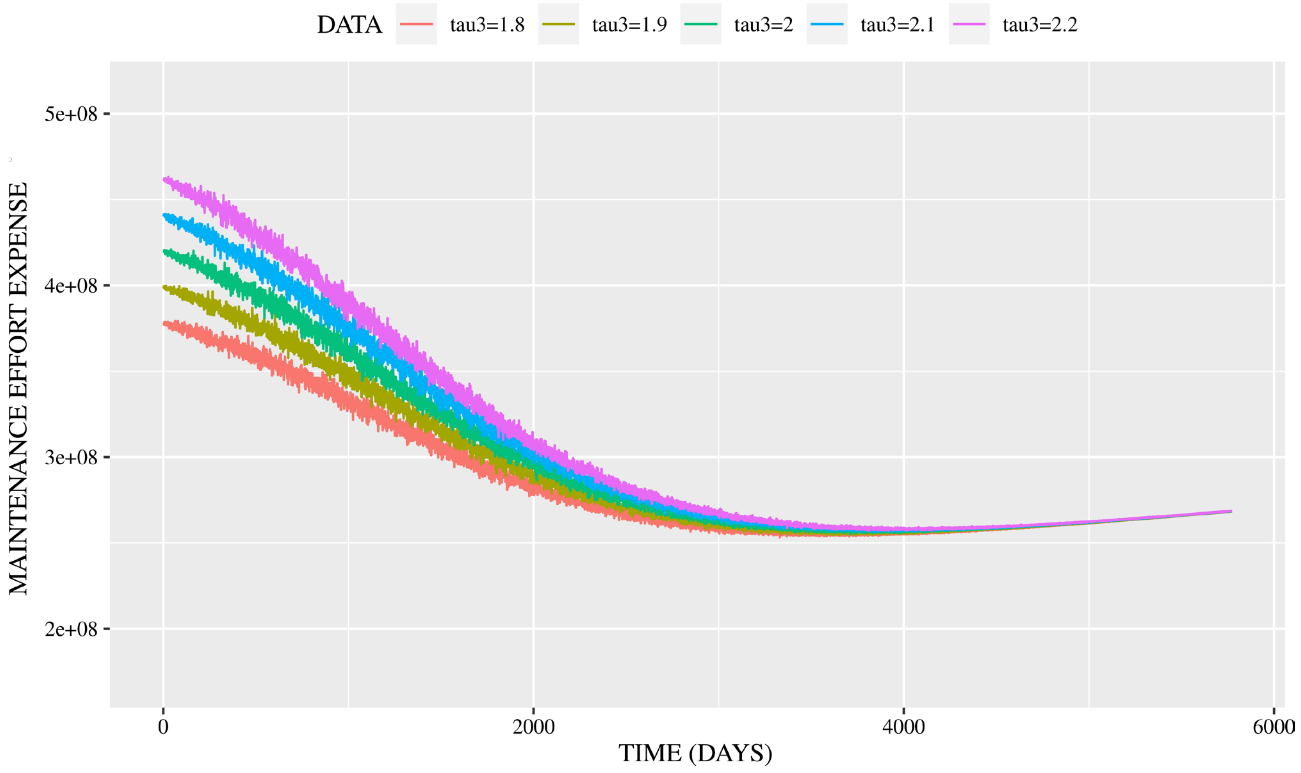

Figure 7. The sensitivity analysis of the sample paths for effort expense parameter τ_3.

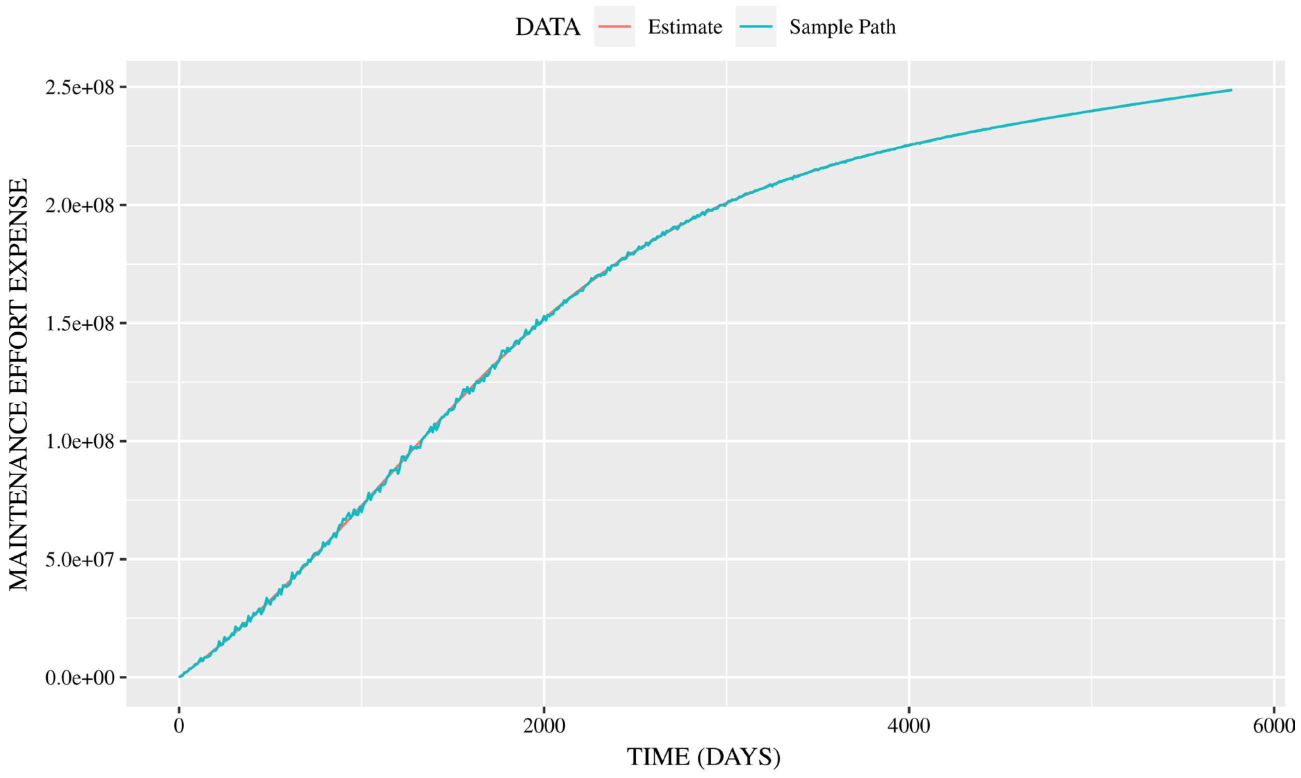

Figure 8. The estimated software effort expense in Equation (5) based on genetic algorithm.

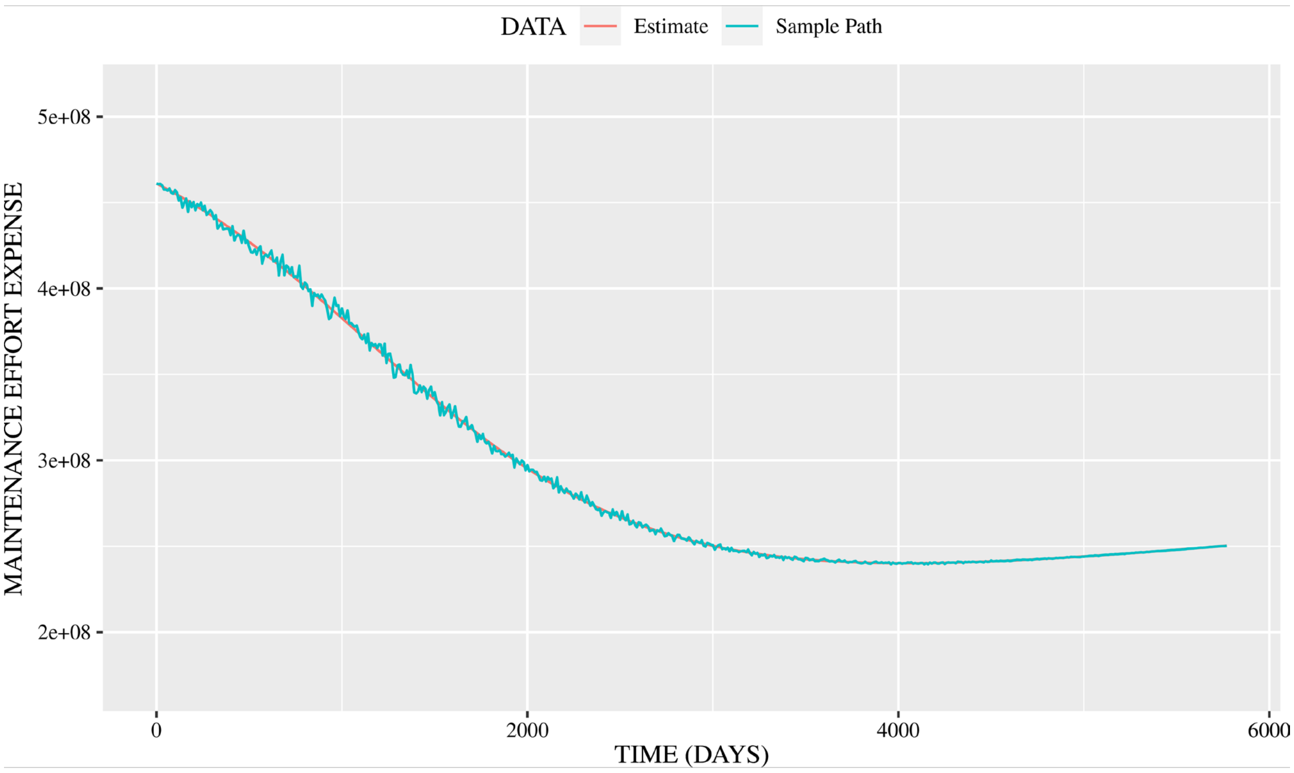

Figure 9. The estimated software effort expense in Equation (6) based on genetic algorithm.

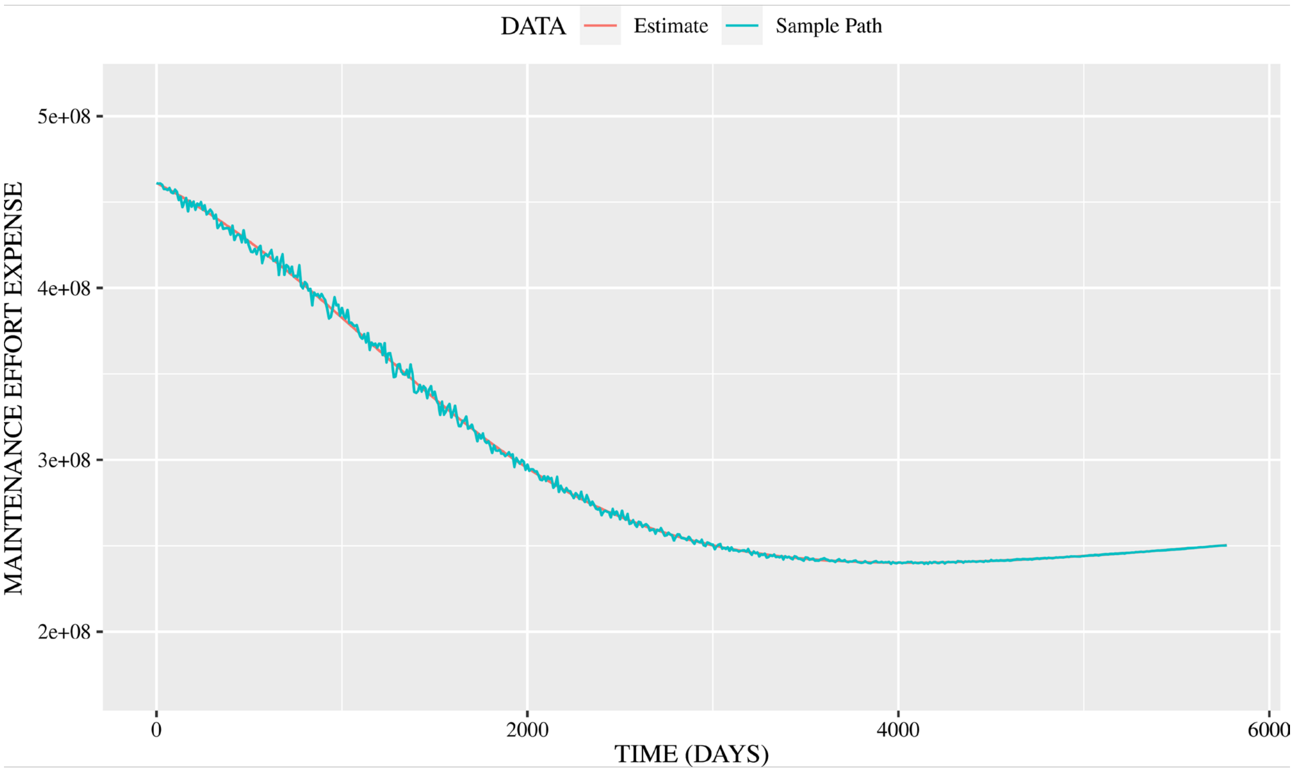

Figure 10. The estimated total software effort expense based on genetic algorithm.

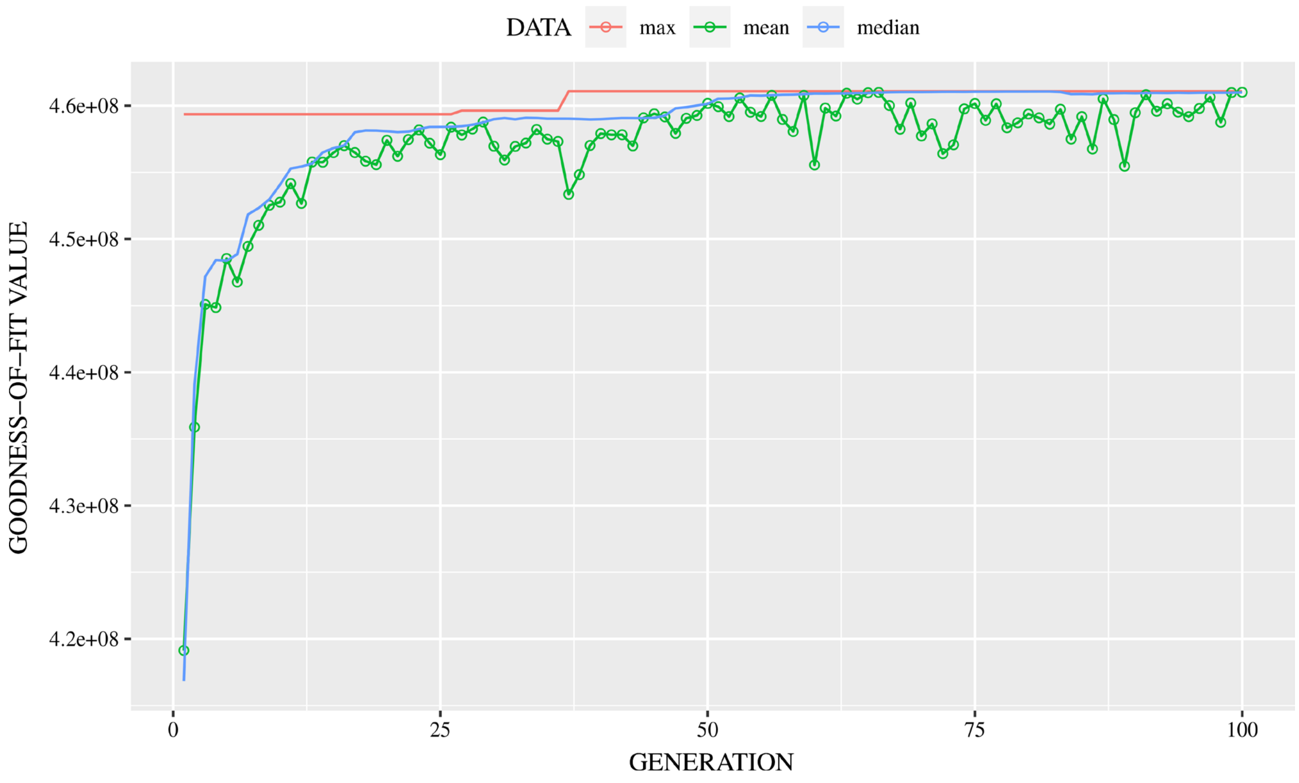

Figure 11. The convergence status based on genetic algorithm.

cost parameters and effort ones have been given as the experimental parameters of software managers in the past. Therefore, we cannot compare the proposed method with the conventional method, because there is no similar method.

- The managers of OSS project can decide the optimal software operation effort by using the estimated effort parameters τ_1, τ_2, and τ_3, respectively. Thereby, the OSS managers can achieve the cost reduction and quick delivery.

Also, several research papers in terms of the optimization algorithms have been proposed in the past [16] [17]. However, these methods are not for the area of software engineering.

From above mentioned, it is difficult to compare the proposed method with the conventional one. Alternatively, we show the comparison results based on several GA's. Figures 10 and Figures 11 are the results based on "L-BFGS-B" [18]. As the comparison results by using the "Nelder-Mead" [19], we show the estimated total software effort based on GA in Figures 12. Similarly, we show the convergence status in Figures 13. Then, the effort parameters are estimated by using Equations (8)-(10) in section 3 as follows:

$$\tau_1 = 1.0362, \tau_2 = 9617.0, \tau_3 = 2.1984.$$

As the comparison results based on two kinds of algorithms, the estimation results based on "Nelder-Mead" is pessimistically estimated from Figure 10 and Figure 12. However, we found that there is almost no difference between "L-BFGS-B" and "Nelder-Mead" in terms of the optimization algorithms in GA.

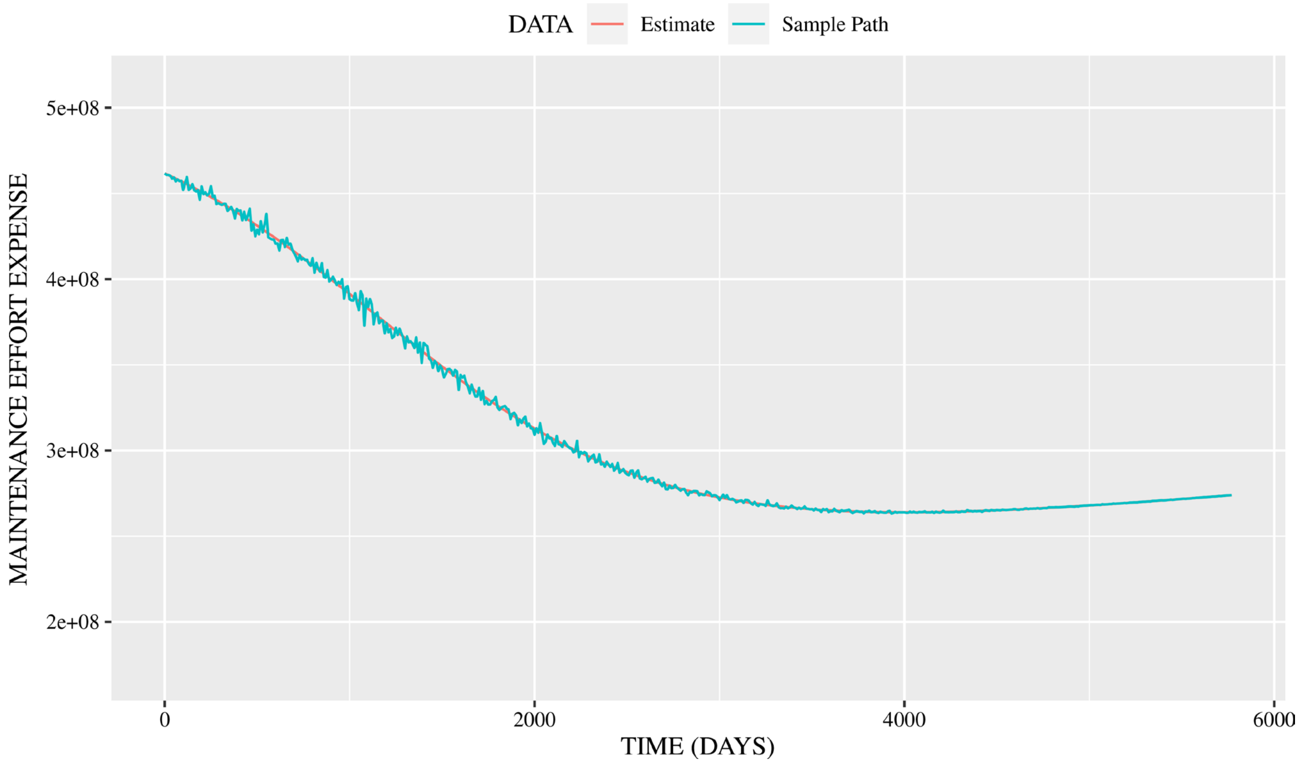

Figure 12. The estimated total software effort expense based on genetic algorithm based on "Nelder-Mead".

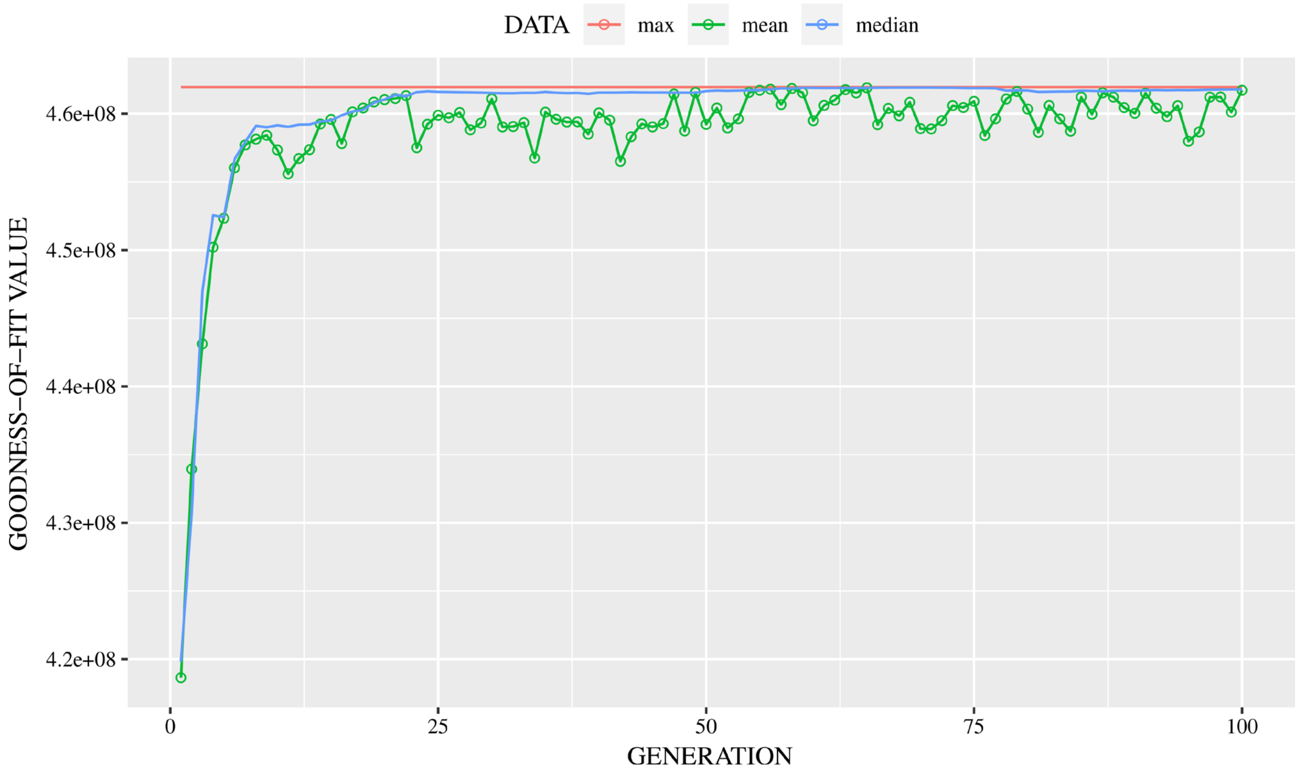

Figure 13. The convergence status based on genetic algorithm based on "Nelder-Mead".

Therefore, the proposed method is stable without regard to the optimization algorithms in GA.

5. Conclusions

We have discussed the optimum maintenance problem based on the genetic algorithm. In particular, we have proposed an estimation method of effort expense parameters included in our effort function. Moreover, we have concretely shown numerical examples of effort optimization. Then, we have found that the OSS managers can comprehend the human resources required before the OSS project in advance.

Finally, we have discussed the contribution of the proposed method. In particular, we have focused on two dimensional Wiener processes model for effort expense estimation. We conclude the advantage points as follows:

1st Point: The proposed stochastic differential equation model can assess OSS projects considering the influence of big data in terms of 3 V model.

2nd Point: The proposed method can search the optimum maintenance time under the complex situation such as Equation (9).

3rd Point: Before the OSS operation, the OSS managers can comprehend the required human resource in advance.

The OSS managers will be able to control the software effort expense from the stand point of 3 V model as big data by using the proposed effort expense optimization analysis.

At present, the cloud and edge computing based on OSS is attracting the most attention. It is difficult to assess the optimum maintenance time, because the cloud and edge computing have the network-based-service and many distributed node servers. As the future study, it will be important to discuss the optimum maintenance problem for the cloud and edge computing service.

Acknowledgements

This work was supported in part by the JSPS KAKENHI Grant No. 20K11799 in Japan.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Ibrahim, I.M., *et al.* (2018) A Robust Generic Multi-Authority Attributes Management System for Cloud Storage Services. *IEEE Transactions on Cloud Computing*, 30 August 2018, 1. https://doi.org/10.1109/TCC.2018.2867871

[2] Ahmad, A.A., *et al.* (2019) Scalability Analysis Comparisons of Cloud-Based Software Services. *Journal of Cloud Computing*: *Advances*, *Systems and Applications*, **8**, Article No. 10.

[3] Taleb, T., Samdanis, K., Mada, B., Flinck, H., Dutta S. and Sabella, D. (2017) On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud

Architecture and Orchestration. *IEEE Communications Surveys & Tutorials*, **19**, 1657-1681. https://doi.org/10.1109/COMST.2017.2705720

[4] Kapur, P.K., Pham, H., Gupta, A. and Jha, P.C. (2011) Software Reliability Assessment with OR Applications. Springer-Verlag, London. https://doi.org/10.1007/978-0-85729-204-9

[5] Yamada, S. and Tamura, Y. (2016) OSS Reliability Measurement and Assessment. Springer International Publishing, Switzerland. https://doi.org/10.1007/978-3-319-31818-9

[6] Norris, J. (2004) Mission-Critical Development with Open Source Software. *IEEE Software Magazine*, **21**, 42-49. https://doi.org/10.1109/MS.2004.1259211

[7] Singh, V.B., Sharma, M. and Pham, H. (2017) Entropy Based Software Reliability Analysis of Multi-Version Open Source Software. *IEEE Transactions on Software Engineering*, **44**, 1207-1223. https://doi.org/10.1109/TSE.2017.2766070

[8] Yamada, S. and Osaki, S. (1985) Cost-Reliability Optimal Software Release Policies for software Systems. *IEEE Transactions on Reliability*, **34**, 422-424. https://doi.org/10.1109/TR.1985.5222222

[9] Khatri, S.K., John S.A. and Majumdar, R. (2016) Quantifying Software Reliability Using Testing Effort. *Proceedings of International Conference on Information Technology* (*InCITe*)*—The Next Generation IT Summit on the Theme—Internet of Things*: *Connect Your Worlds*, Noida, 6-7 October 2016, 23-26. https://doi.org/10.1109/INCITE.2016.7857582

[10] Lance, F. and Swapna, S.G. (2008) Software Reliability Models Incorporating Testing Effort. *OPSEARCH*, **45**, 351-368. https://doi.org/10.1007/BF03398825

[11] Kapur, P.K., Gupta, A. and Jha, P. (2007) Reliability Analysis of Project and Product Type Software in Operational Phase Incorporating the Effect of Fault Removal Efficiency. *International Journal of Reliability*, *Quality and Safety Engineering*, **14**, 219-240. https://doi.org/10.1142/S021853930700260X

[12] Tamura, Y. and Yamada, S. (2019) Maintenance Effort Management Based on Double Jump Diffusion Model for OSS Project. *Annals of Operations Research*, 1-16. https://doi.org/10.1007/s10479-019-03170-w

[13] Arnold, L. (1974) Stochastic Differential Equations-Theory and Applications. John Wiley & Sons, New York.

[14] Holland, J.H. (1975) Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor.

[15] The Apache Software Foundation. The Apache HTTP Server Project. http://httpd.apache.org/

[16] Nayak, S.C. and Misra, B.B. (2020) Extreme Learning with Chemical Reaction Optimization for Stock Volatility Prediction. *Financial Innovation*, **6**, Article No. 16. https://doi.org/10.1186/s40854-020-00177-2

[17] Balderas, F., Fernandez, E., Gomez-Santillan, C., Rangel-Valdez, N. and Cruz, L. (2019) An Interval-Based Approach for Evolutionary Multi-Objective Optimization of Project Portfolios. *International Journal of Information Technology & Decision Making*, **18**, 1317-1358. https://doi.org/10.1142/S021962201950024X

[18] Byrd, R.H., Lu, P., Nocedal, J. and Zhu, C. (1995) A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific Computing*, **16**, 1190-1208. https://doi.org/10.1137/0916069

[19] Nash, J.C. (1990) Compact Numerical Methods for Computers, Linear Algebra and Function Minimisation. Adam Hilger, Bristol.