*Advances in Intelligent Systems Research, volume 169*
VIth International Workshop 'Critical Infrastructures: Contingency Management, Intelligent, Agent-Based, Cloud Computing and Cyber Security' (IWCI 2019)

# Simulation of Software Security Tests by Soft Computational Methods

Alexey S. Markov
*Bauman Moscow State Technical University*
Moscow, Russia
a.markov@bmstu.ru

Georgii A. Markov
*Jet Infosystems, JSC*
Moscow, Russia
gm@cnpo.ru

Valentin L. Tsirlov
*Bauman Moscow State Technical* University
Moscow, Russia
v.tsirlov@bmstu.ru

***Abstract* — The worth of the using of software performance reliability assessment models at early testing stages is shown. The characteristic property of nonmonotonic increasing of performance reliability of current software systems is underscored. Ways of enhancing adequacy and fidelity of software testing simulation are exemplified. The conditions of using of soft computing for assessing the reliability and security of information systems software are shown.**

***Keywords* — *software performance reliability, software testing, reliability growth models, debugging models, fuzzy models, artificial neural networks, soft computing***

## I. INTRODUCTION

Approval of mandatory requirements for formalizing the results of compliance assessment of high-confidence information protection software [1] lent relevance to the use of mathematical models for assessing IS software performance reliability and safety. Although studies into mathematical models for software reliability assessment were first undertaken back in the second half of the last century [2, 3], these issues remain relevant nowadays [4-16]. All this relates both to software manufacturing technologies (for example, open source software) and to newly introduced international standards in the software engineering and information security area (ISO/IEC 15408, ISO/IEC 33001, IEC 61508, IEC 61511 etc.) [7, 17, 18]. It should be understood that the concept of reliability is equivalent to that of software safety, if it is assumed that all detected defects (errors) and vulnerabilities are not strictly identified as deliberate. Therefore, the software testing models under review herein will be conventionally termed as reliability models.

As is well known, development, selection and synthesis of mathematical models are directly dependent on the statistics gathering subsystem implemented in the software life cycle. This article investigates the possibility and applications of soft computing as part of mathematical test modeling and software maintenance.

## II. DEBUGGING MODEL

There are a few taxonomies of mathematical software reliability models known at present [2, 19-22]. We believe that, from the applied perspective, it is convenient to use the classification relating to the objectives of software product/system testing stages [21], which allows 4 types of mathematical models to be singled out:

- Test planning models taking into account software complexity (program complexity models);
- Debugging models oriented toward multiple updates and input data domain coverage (data-domains models);
- Time-dependent reliability growth models intended for trial and production use (time-domains models);
- Test confidence models.

Debugging models are the most sensible to the statistics gathering system, as it is initial testing (debugging) stages that are characterized by multiple changes in test object versions, including in various media and platforms. The first offered models of the specified class are Nelson and LaPadula models plus their modifications [3, 21-26] which imply a strictly monotonous growth of software reliability. At present, this is not always consistent with current multiversion programming technologies. To rule out the above drawback, the authors gave a thorough study to software implementation of probabilistic reliability growth model allowing some additional positive and negative debugging factors to be taken into account, namely [21]:

$$P_u = P_\infty - (P_\infty - P_0)\prod_{j=1}^{u}(1 - A_j / P_\infty), \quad (1)$$

where: $P_0$ – initial reliability level, $P_\infty = \frac{A_j}{A_j + B_j}$ - ultimate reliability level, $0 \le P_0 < P_\infty \le 1$, $u$ – number of completed modifications; $A_j$ - modification "effectiveness" factor characterizing an error probability decrease due to the *j*-th modification; $B_j$ -

modification "negativeness" factor characterizing reliability degradation caused by the *j*-th modification

Depending on the available statistics, the test modeling/software debugging evolution may take, for example, the following paths:

- Updating a probabilistic model (1) parameters depending on the significance of a target software update class;
- Updating a probabilistic model (1) parameters on the code update complexity level;
- Bayes modifications of a model when updating the stochastic nature of its parameters;
- Developing an interval model based on the fuzzy sets theory machinery;
- Developing a model by means of artificial neural network technologies etc.

The last two modeling options are considered to pertain to soft computing, which will be discussed a bit later. Because Bayes modifications make model visualization (simplicity) parameter determination much more difficult, they were left outside this article. Yet, the above theoretical approach is considered in literature [27-30]. Let us touch briefly upon the issues of model (1) refinement by modifying model parameters.

## III. Analytical Model Refinement

As far as the updating of a probabilistic model's parameters (1) is concerned, the most graphical model is the bigeminal debugging one taking account of opposite software modification classes, namely: correcting an error and introducing a new software functional [21].

$$P_u = P_\infty - (P_\infty - P_0)\prod_{j=1}^{u}\left(1 - \sum_{i=1}^{2} a_i k_{ij} / P_\infty\right), \quad (2)$$

where: $a_1$ - effectiveness factor of software modification for error correction, $a_2$ - effectiveness factor of software modification for adding functional capabilities, $k_{ij}$ - scope of the *j*-th modification aimed at correction or update (equal to 1 by default).

The bigeminal model (2) depends on 4 parameters $(P_0, P_\infty, a_1, a_2)$ easy to calculate, for example, using the maximum likelihood method [21]. No restrictions are imposed on the domain of defining the modification scope factor $k_{ij}$ to allow consistency and completeness of describing the testing and debugging process. Thus, where statistics on source code modification are available, it is easy to utilize the known complexity metrics. For machine-oriented and procedural (also partially object-oriented) languages, it is easily implemented by making use of heuristic and statistical complexity metrics (Halstead, McCabe, Chepen etc.). In case of visual programming systems, the introduction of popular complexity metrics may sometimes fall short of weary anticipations. Ultimately, they may be simply reduced to length or scope of modified fragments.

Practical modeling experience showed that adequacy and fidelity could be enhanced by taking into account the phenomenon of nonmonotonic growth of multiversion software reliability and diverse wave errors. Graphically, the software reliability variation process is represented by a step reliability growth function (Figure 1).

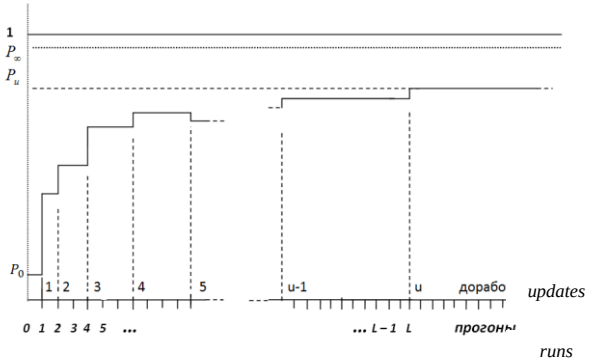



Fig.1 Reliability degree variation after updates

## IV. Use of Soft Computing

Soft computing is normally undertaken in apparent cases of available statistics on testing and debugging software products and systems. As mentioned before, the most popular approaches include the use of artificial neural system technologies [19, 31-33] and fuzzy set apparatus [34-37]. Given below are examples of developing such models.

### *A. 4.1. Using neural network technologies*

A transition to neural networks stems from the difficulty of accounting updates in multiversion software systems (in particular, open source systems where programmers are distributed by time, style, technology and qualification). As a result, there are some regularities unclear at initial stages. One must realize that neural technologies can be implemented providing there are statistics representative enough in volume.

A package of computer modeling in Neuroshell2 (Pro) was chosen by a computer experiment to verify this approach [32]. As part of the study, the authors checked if modeling was possible by using allowed topologies of neural networks and selecting the optimum number of their parameters (number of neurons, layers, link types etc.). Source data and the computer experiment outcomes are presented in figures 2-5.

The study found even a simple 4-layer neural network and a network with bypass connection to have coped with the task set [31]. It should be realized that applying neural network technologies requires support by the relevant data gathering system and, naturally, accumulation of highly representative statistics.

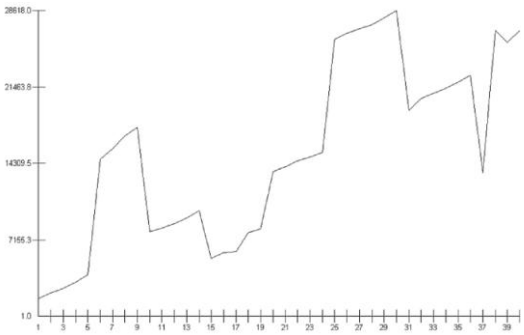
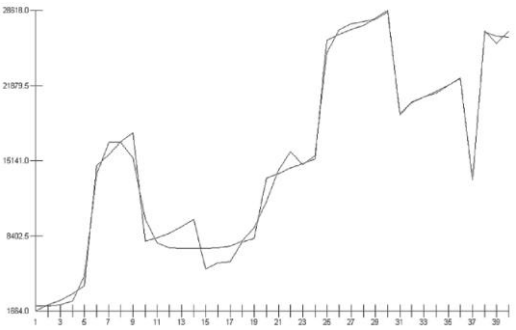


Fig. 2 Source data (number of successful tests)



Fig, 3 Using a simple neural network

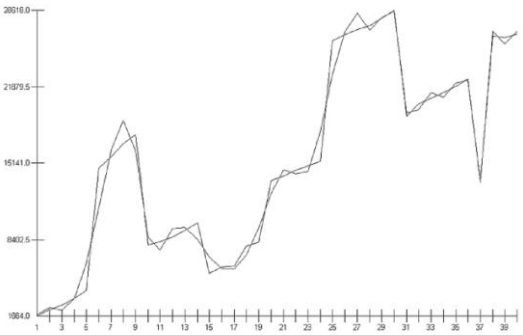


Fig. 4 Using a network with bypass connections

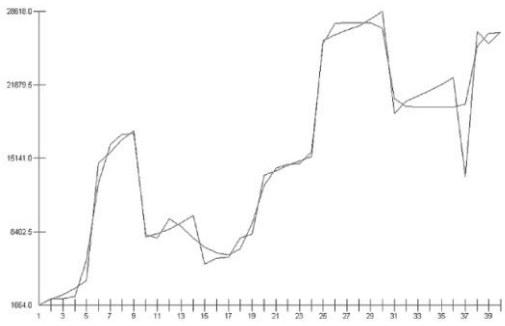


Fig. 5 Using recurrent networks with negative feedback

### *B. 4.2. Using the fuzzy set theory apparatus*

Using the fuzzy set theory apparatus is convenient when there are no requirements for high accuracy of decision making assessment and the degree of uncertainty of statistics can be accounted by fuzzy expert estimates. Thus, by slightly simplifying the model (1):

$$P_u = P_\infty - (P_\infty - P_0)(1 - \frac{a}{P_\infty})^u,$$

it is convenient to pass on to the fuzzy model:

$$\boldsymbol{P} = \{(P_u, \mu_P(P_u))\},$$

where: $P_u$ – software reliability degree in case of $u$-modifications; $\mu_P(P_u)$ - membership function.

Showing the confidence level $\alpha$ allows the desired interval fuzzy model to be obtained in general terms:

$$\boldsymbol{P} = \{P_m \mid \mu_M(m) \geq \alpha\}.$$

The fuzzy model development is based on determining a range of fuzzy sets, such as:

- A fuzzy set $\boldsymbol{A} = \{(\mathrm{x}_i, \mu_A(\mathrm{x}_i))\}$ representing a totality of ordered pairs of software modifications – the ground set $X$ and the membership function $\mu_A(\mathrm{x}_i)$ characterizing the availability of software modification (update).
- A fuzzy set of accountable $\boldsymbol{U} = \{(u, \mu_U(u))\}$, where $\mu_U(u)$ - a membership function meaning the degree of confidence that the number of accountable modifications is equal to $u$.

This approach, including membership function obtaining techniques, is described in detail in [21].

## V. CONCLUSION

Notwithstanding the long-term investigations in the software testing, reliability and quality field, introduction and development of mathematical testing models remains sought-after. This, in particular, is specified in some international standards concerning both procedural guidelines as such and software safe life cycle procedures. At present, there is a representative set of mathematical models whose selection and synthesis depend on specific situations, including the reliability assessment of software products based on current and advanced software manufacturing systems.

The work shows the possibility of using soft computing in apparent cases of available statistics on testing current software systems. The results obtained using soft computing methods can be applied as initial when calculating parameters of analytical models.

This study is within the scope of development and introduction of a range of new standards harmonized with GOST R 56939, a national standard for safe software development [38].

## REFERENCES

[1] A. Barabanov, and A. Markov, "Modern trends in the regulatory framework of the information security compliance assessment in Russia based on common criteria," in Proceedings of the 8th International Conference on Security of Information and Networks (Sochi, Russian Federation, September 08-10, 2015), SIN '15, ACM New York, 2015, pp. 30-33. DOI: 10.1145/2799979.2799980.

[2] S.S. Gokhale, P.N. Marinos, and K.S. Trivedi, "Important milestones in software reliability modeling," in Proc. of

Software Engineering and Knowledge Engineering (SEKE 96), Lake Tahoe, NV, 1996, pp. 345-352.

[3] T.A. Teyer, M. Lipow, and E.C. Nelson, Software Reliability. A Study of Large Project Reality. TRW Systems and Energy, 1978. 326 p.

[4] B. Anniprincy, and S. Sridhar, "Prediction of software reliability using cobb-douglas model in srgm," Journal of Theoretical and Applied Information Technology, vol 62, no 2, pp. 355-363, 2014.

[5] V.P. Bubnov, and S.A. Sergeev, "Non-stationary models of a local server of the automated system for monitoring artificial structures,". SPIIRAS Proceedings, no 45, pp. 102–115, 2016. DOI: 10.15622/sp.45.6.

[6] D. Gaskova, and A. Massel, "Intelligent System for Risk Identification of Cybersecurity Violations in Energy Facility", in Proceedings of the:2018 3rd Russian-Pacific Conference on Computer Technology and Applications (Vladivostok, Russia, August 18-25, 2018), RPC, IEEE, 2018, pp 1-5. DOI: 10.1109/RPC.2018.8482229.

[7] A.I. Danilov, A.D. Khomonenko, and A.A. Danilov, "Dynamic software testing models," in Proceedings of International Conference on Soft Computing and Measurements (SCM 2015), 19-21 May 2015, St. Petersburg, Russia: IEEE, 2015, pp. 72-74. DOI: 10.1109/SCM.2015.7190414.

[8] A.N. Ivutin, E.V. Larkin, and D.A. Perepelkin, "Software errors and reliability of embedded software," in 2016 IEEE Conference on Quality Management, Transport and Information Security, Information Technologies (IT&MQ&IS), 4-11 Oct. 2016, Nalchik, Russia: IEEE, 2016, pp. 69-71. DOI: 10.1109/ITMQIS.2016.7751926.

[9] A. Kostogryzov, "Modeling software tools complex for evaluation of information systems operation quality," Lecture Notes in Computer Science, vol. 2052, pp. 90-101, 2001. DOI: 10.1007/3-540-45116-1_12.

[10] V.G. Krymsky, and I.V. Ivanov, "Application of interval-valued probabilities and unified scheme of non-homogeneous poisson process models to software failure prognostics," in L. Podofillini, and etc (editors), Safety and Reliability of Complex Engineered Systems: ESREL 2015, CRC Press, 2015. p. 2403-2411.

[11] V.A. Smagin, A.N. Novikov, and S.Yu. Smagin, "A probabilistic model of the control of technical systems," Automatic Control and Computer Sciences, vol. 44, no 6, pp. 324-329, 2010. DOI: 10.3103/S0146411610060027.

[12] R. Subburaj, Software Reliability Engineering, McGraw Hill Education, 2014. 458 p.

[13] Y. Tamura, and S. Yamada, "Cost optimization based on decision-making and reliability modeling for big data on cloud computing." Communications in Dependability and Quality Management, vol. 18, no 4, pp. 5-19. 2015.

[14] S. Yamada, Software Reliability Modeling: Fundamentals and Applications. Springer Japan, 2014, 90 p. DOI: 10.1007/978-4-431-54565-1.

[15] P. Zeephongsekul, C.L. Jayasinghe, and L. Fiondella, "Vidhyashree nagaraju maximum-likelihood estimation of parameters of nhpp software reliability models using expectation conditional maximization algorithm," IEEE Transactions on Reliability, vol. 65, no 3, pp. 1571-1583, 2016. DOI: 10.1109/TR.2016.2570557.

[16] C. Zhao, J. Qiu, G. Liu, and K. Lv, "Planning, tracking and projecting method for testability growth based on in time correction," in Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability, vol. 230, no 2, pp. 228-236, 2015. DOI: 10.1177/1748006X15626026.

[17] G. Reber, K. Malmquist, and A. Shcherbakov, "Mapping the application security terrain," Voprosy kiberbezopasnosti [Cybersecurity issues], no 1(2), pp. 36-39, 2014. DOI: 10.21681/2311-3456-2014-2-36-39.

[18] A. Barabanov, A. Markov, A. Fadin, V. Tsirlov, I. Shakhalov, "Synthesis of secure software development controls," in Proceedings of the 8th International Conference on Security of Information and Networks (Sochi, Russian Federation, September 08-10, 2015). SIN '15. ACM New York, pp. 93-97, 2015. DOI: 10.1145/2799979.2799998.

[19] K.S. Kaswan, S. Choudhary, and K. Sharma, "Software reliability modeling using soft computing techniques: critical review," J Inform Tech Softw Eng., vol 5, no 144, pp. 1-9, 2015. DOI: 10.4172/2165-7866.1000144.

[20] D. Maevsky, V. Kharchenko, M. Kolisnyk, and E. Maevskaya, "Software reliability models and assessment techniques review: classification issues," in 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), 21-23 Sept. 2017, Bucharest. Romania: IEEE, 2017, pp. 894–899, 2017. DOI: 10.1109/IDAACS.2017.8095216.

[21] A. Markov, A. Barabanov, and V. Tsirlov, "Models for testing modifiable systems," in book: A. Kostogryzov, editor, Probabilistic Modeling in System Engineering. IntechOpen, 2018, Chapter 7, pp. 147-168. DOI: 10.5772/intechopen.75126.

[22] A. Markov, A. Barabanov, and V. Tsirlov, "Periodic Monitoring and Recovery of Resources in Information Systems," in book: A. Kostogryzov, editor, Probabilistic Modeling in System Engineering. IntechOpen, 2018, Chapter 10, pp. 213-231. DOI: 10.5772/intechopen.75232.

[23] B. Andersson, and M. Persson, Software Reliability Prediction - An Evaluation of a Novel Technique. SEBIT, 2004. 32 p.

[24] P.K. Kapur, H. Pham, A. Gupta, and P.C. Jha, Software Reliability Assessment with OR Applications. London: Springer-Verlag, 2013, 548 p. DOI: 10.1007/978-0-85729-204-9.

[25] J. Tian, Software Quality Engineering: Testing, Quality Assurance and Quantifiable Improvement. Wiley-IEEE CSP, 2005, 440 p.

[26] M. Xie, Y.-S. Dai, K.L. Poh, Computing Systems Reliability. Models and Analysis. Kluwer Academic Publishers, 2004, 293 p. DOI: 10.1007/b100619.

[27] R. Rana, M. Staron, C. Berger, J. Hansson, M. Nilsson, and W. Meding, "Analyzing defect inflow distribution and applying bayesian inference method for software defect prediction in large software projects," Journal of Systems and Software, vol. 117, pp. 229-244, 2016. DOI: 10.1016/j.jss.2014.08.033.

[28] H.A. Stieber, "Estimating the total number of software faults reliability models and mutation testing a bayesian approach," in 2015 IEEE 39th Annual Computer Software and Applications Conference, 1-5 July 2015, Taichung, Taiwan: IEEE, pp. 423-426, 2015. DOI: 10.1109/COMPSAC.2015.180.

[29] L.V. Utkin, S.I. Zatenko, and F.P.A. Coolen, "New interval bayesian models for software reliability based on non-homogeneous poisson processes," Automation and Remote Control, vol. 71, no 5, pp. 935-944, 2010. DOI: 10.1134/S0005117910050218.

[30] L.J. Wang, Q.P. Hu, and M. Xie, "Bayesian analysis for nhpp-based software fault detection and correction processes," in 2015 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), IEEE, pp. 1046-1050, 2015. DOI: 10.1109/IEEM.2015.7385808.

[31] M. Bisi, and N.K. Goyal. Artificial Neural Network Applications for Software Reliability Prediction. Wiley-Scrivener, 312 p.

[32] G. Markov, "The experiment on the use of neural network technology for test open software," Voprosy kiberbezopasnosti [Cybersecurity issues], no 2, pp. 47-52, 2013. DOI: 10.21681/2311-3456-2013-2-47-52.

[33] Yu.I. Starodubtsev, E.V. Grechishnikov, and D.V. Komolov, "Use of neural networks to ensure stability of communication networks in conditions of external impacts," Telecommunications and Radio Engineering, vol. 70, no 14, pp.. 1263-1275, 2011.

[34] G. Junhong, Y. Xiaozong, and L. Hongwei, "Software reliability nonlinear modeling and its fuzzy evaluation," in 4th WSEAS Int. Conf. on Non-Linear Analysis, Non-Linear Systems and Chaos (NOLASC'05), 27-29 October 2005, Sofia, Bulgaria: ACM, pp. 49-54, 2005.

[35] R. Kumar, K. Khatter, and A. Kalia, "Measuring software reliability: a fuzzy model," in ACM SIGSOFT Software Engineering Notes, vol. 36, no 6, pp. 1-6, 2011. DOI: 10.1145/2047414.2047425.

[36] E.G. Vorobiev, S.A. Petrenko, I.V. Kovaleva, and I.K. Abrosimov, "Analysis of computer security incidents using fuzzy logic," in Proceedings of the 20th IEEE International Conference on Soft Computing and Measurements (24-26 May 2017, St. Petersburg, Russia). SCM 2017, pp. 369 – 371, 2017. DOI: 10.1109/SCM.2017.7970587.

[37] E.G. Vorobiev, S.A. Petrenko, I.V. Kovaleva, and I.K. Abrosimov, "Organization of the entrusted calculations in crucial objects of informatization under uncertainty," in Proceedings of the 20th IEEE International Conference on Soft Computing and Measurements (24-26 May 2017, St. Petersburg, Russia). SCM 2017, pp. 299-300, 2017. DOI: 10.1109/SCM.2017.7970566.

[38] A. Barabanov, A. Markov, V. Tsirlov, "Procedure for substantiated development of measures to design secure software for automated process control systems," in Proceedings of the 12th International Siberian Conference on Control and Communications (Moscow, Russia, May 12-14, 2016). SIBCON 2016. IEEE, 7491660, pp. 1-4, 2016. DOI: 10.1109/SIBCON.2016.7491660.