Визначено, що існує ряд специфічних особливостей, пов'язаних з можливістю негативних маніпуляцій програмним продуктом. Розроблено графоаналітичну модель алгоритму тестування безпеки програмного забезпечення. Отримано співвідношення для розрахунку статистичних характеристик випадкового часу виконання даного алгоритму. Модель можна використовувати для дослідження етапів тестування безпеки програмного забезпечення. Застосування моделі дозволить знизити уразливість IT-проекту в цілому

Ключові слова: тестування безпеки, графоаналітична модель, напівмарківський процес, перетворення Лапласа, виробляюча функція

Определено, что существует ряд специфических особенностей, связанных с возможностью негативных манипуляций программным продуктом. Разработана графоаналитическая модель алгоритма тестирования безопасности программного обеспечения. Получены соотношения для расчета статистических характеристик случайного времени выполнения данного алгоритма. Модель можно использовать для исследования этапов тестирования безопасности программного обеспечения. Применение модели позволит снизить уязвимость IT-проекта в целом

Ключевые слова: тестирование безопасности, графоаналитическая модель, полумарковский процесс, преобразование Лапласа, производящая функция

UDC 004.415.53: 519.711
DOI: 10.15587/1729-4061.2018.127210

DEVELOPMENT OF GRAPHIC-ANALYTICAL MODELS FOR THE SOFTWARE SECURITY TESTING ALGORITHM

S. Semenov
Doctor of Technical Sciences, Senior Researcher
Department of Computer Science and Programming*
E-mail: s_semenov@ukr.net
O. Sira
Doctor of Technical Sciences, Professor
Department of distributed information systems and cloud technologies*
E-mail: topology@ukr.net
N. Kuchuk
PhD
Department of Theoretical and Applied Systems Engineering
V. N. Karazin Kharkiv National University
Svobody sq., 4, Kharkiv, Ukraine, 61022
E-mail: nina_kuchuk@ukr.net
*National Technical University «Kharkiv Polytechnic Institute»
Kyrpychova str., 2, Kharkiv, Ukraine, 61002

1. Introduction

The software development practice and the trends in working out methodologies for managing IT projects show that the software testing issues draw the ever-growing attention. The problems of software testing are of especial importance for present-day computers and computerized systems of critical application (CSCA).

The software development experience shows that security testing is one of the most complicated and important types of testing. It becomes especially important in the context of insistent cyberthreats of various kinds associated with vulnerability of software codes and growth of risks of negative impact of software failures on the overall of the CSCA operation process.

According to expert data [1, 2], the time of testing software security can take up to 35 % of the total testing time. However, these time costs are not always justified. Besides, such a long time does not guarantee sufficient completeness of tests and the specified level of the CSCA software security. Therefore, nowadays, evaluation of quality of the software security tests requires constant care.

The check based on the models of corresponding algorithms is the most acceptable approach to assessing test quality. However, the problem of model conformity to the process under consideration becomes of primary importance. That is why the issue of developing alternative mathematical models of the testing algorithm that meet conformity requirements as considered in this paper is topical.

2. Literature review and problem statement

Quality of the software security tests is discussed in works of many contemporary scientists. It is shown in [3] that the best result of testing security of the CSCA software can be achieved by an utmost approach to and complying with the points of the software product specification. Authors of paper [4] bring up a question of necessity of modeling the software security tests. At present, there are methods developed for generating functional and other tests that are as close as possible to the specification requirements [5]. Also, theoretically grounded ways of improving effectiveness of software testing are offered in a number of studies. For example, features of software for mobile communication networks are taken into account in [6]. Issues of testing quality evaluation for boundedly nondeterministic systems are considered in [7]. However, most of these works do not take into

consideration the specifics and experience of testing security of the CSCA software. Therefore, the issues of mathematical formalization of the testing process and, accordingly, development of the mathematical model of security algorithm of the CSCA software testing remain unresolved. At the same time, it is of fundamental importance to take into account features of the software security testing process. As shown in [8], one of the features of the software security testing process is non-specificity of the test techniques in the course of solving the problems posed. This feature is determined by the fact that the tester most often plays the role of a hacker and starts various non-standard manipulations with the software product. Such manipulations include, for example, the following:

– attempts to mouse out the password by using external means;

– attacking systems by using special utilities that analyze security;

– DOS attacks;

– intentional embedding of errors with further penetration into the system in the course of its recovery;

– browsing unclassified data in the hope of finding a key to log into the system.

It is these non-specific techniques in combination with an accumulated knowledge of main types of the CSCA software vulnerability that put the test cases in a fuzzy conformance with the existing software product specification. Such a fuzzy conformance of the test cases with the specification requirements is called "conformality" [9].

In a semantic description of the simulated system, it is necessary to take into cpnsideration the fact that observations of the tested software product can be divided into two types: observations at a specified impact of the action performed by the test and observations of the action failure.

Thus, semantics of interaction between the CSCA software security test and the software can be defined by the alphabet of external actions. Strategies for implementing such an interaction are described in [10]. Features of the semantic description of the models for CSCA are given in [11].

When setting the problem of mathematical formalization of the CSCA software testing algorithm, the following factors should be considered [12]:

A. Interaction of the tester with the software tested should be safe for the CSCA.

B. Interaction with the software product allows one to get information on its capabilities. To this end, it is possible to introduce some restrictions to the interaction semantics, specification, and implementation. Given the availability of additional (nonspecific) test capabilities, these conditions appear to be sufficient to provide a certain level of software security in a finite time.

C. Specification, semantics, and implementation of the functional are finite.

D. Repetition of the same test action in the same state of the software can bring about behavior variants (the condition of nondeterminism). For completeness of testing, it is necessary to check all implementation transitions given in the specification. To do this, the mathematical model must meet a number of conditions:

– adequate mathematical description of the procedure of the implementation restart (restarting the test) in some of its states (the presence of restart transitions leading to the initial state which are taken into account in determining the tight coupling);

– stabilization of the initial state of implementation or at least formalization of one state achievable in the safe specification route;

– separation of complete and partial checks in the process of security testing;

– visually understandable representation of the testing process.

E. The model being developed should adequately reflect the testing procedure.

F. The constructed mathematical model of the testing algorithm should ensure calculation of statistical characteristics of multi-iteration testing procedures.

The graphic-analytical models of complicated systems satisfy the formulated requirements.

3. The aim and objectives of the study

The study objective was to develop alternative mathematical models of the testing algorithm that meet the adequacy requirements and provide solution of the problem of estimating the algorithm effectiveness.

To achieve the objective, the following tasks were set:

– to develop a mathematical model of the testing algorithm based on semi-Markov graphs;

– to develop a mathematical model of the testing algorithm based on the method of probability-time graphs.

4. Mathematical model of the testing algorithm based on the semi-Markov graphs

Conventional technologies for analyzing the semi-Markov systems are limited to calculation of the final distribution of probabilities of the system states using the formulas:

$$P_i = \frac{\pi_j \overline{T}_j}{\sum_{j=1}^{n} \pi_j \overline{T}_j},\ i = 1,2,..,n,$$

where n is the number of probable states of the system;

$$\overline{T}_j = \int_0^{\infty} \left(1 - F_i(t)\right) dt = \int_0^{\infty} \left(1 - \sum_{j=1}^{n} P_{ij} F_{ij}(t)\right) \mathrm{d}t$$

is the mean time of residence in the state i before leaving; π_i is the stationary probability of residence in the state i, i=1, 2,...,n, the set of which is sought by solving a vector-matrix equation

$$(\pi_1, \pi_2, .., \pi_n) = (\pi_1, \pi_2, .., \pi_n) \begin{pmatrix} P_{11} & P_{12} & .. & P_{1n} \\ P_{21} & P_{22} & .. & P_{2n} \\ .. & .. & .. & .. \\ P_{n1} & P_{n2} & .. & P_{nn} \end{pmatrix}.$$

The final probabilities are found from solution of the given system. However, a deeper investigation of the process is necessary rather often, for example, finding the law of distribution of the time interval before falling into any state of the system. To solve the corresponding problems, the apparatus of interval-transition probabilities is used.

As is well known, the semi-Markov process (SMP), differs from the Markov process in that the law of distribution

of the time of residence in each of the states is not necessarily exponential but can be arbitrary.

There are several ways of specifying semi-Markov processes [13, 14]. At first, let us consider the method the least demanding in terms of the amount of information used. Assume that the SMP is defined if the following is given:

1) the set E of possible states and the process transitions;

2) the matrix $\left(Q_{ij}(t)\right)$ of independent functions of distribution of the time of the process residence in the i-th state before transition to the j-th state, $i \in E$, $j \in E$;

3) the initial state of the process at time t=0. $\left(Q_{ij}(t)\right)$ is the function of distribution of the length of residence in i before transition to j provided that there is only one transition to this state. If t_{ij} is the random duration of residence in i before going to another, the only state j, then $Q_{ij}(t) = P\left(t_{ij} < t\right)$.

In accordance with the functions $\left(Q_{ij}(t)\right)$, i, $j \in E$, there exist random time instants t_{ij} of transitions from state i to all states j adjacent to it but only one of them, namely that corresponding to the shortest duration of residence in i is realized, i. e.

$$t_i = \min_{j \in E} t_{ij}. \tag{1}$$

It follows that the probability $p_{ij}(t)$ of transition from state i to state j in time t is the probability that there will be no transition to some other state and that at the moment t, transition to just the j-th state will occur during this time. The probability of transition from i to j in a vicinity of time instant τ is equal to $dQ_{ij}(\tau)$. Then

$$p_{ij}(t) = \int_0^t \prod_{k \neq j} \left(1 - Q_{ik}(\tau)\right) dQ_{ij}(\tau), \quad i \neq j. \tag{2}$$

In accordance with the definition,

$$p_{ij}(t) = P(\xi(t) = j, \ t_{ij} < t \,|\, \xi(0) = i), \quad i \neq j.$$

The function $p_{ij}(t)$ unlike $Q_{ij}(t)$ is not a distribution function since $p_{ij}(\infty) \leq 1$. The set of functions $p_{ij}(t)$, i, $j \in E$ together with the initial state also uniquely specify the SMP.

The probability $p_{ij}(\infty)$ of transition from i to j in an unlimited time is

$$p_{ij} = p_{ij}(\infty) = \int_0^\infty \prod_{k \neq i} \left(1 - Q_{ik}(\tau)\right) dQ_{ij}(\tau) \tag{3}$$

and determines the probability of transition of the Markov chain embedded in the SMP. The embedded Markov chain (IMC) is generated by the SMP if one is only interested in the moments of transition from one state to another. Therefore,

$$p_{ij} = P\left(\xi_{n+1} = j \,|\, \xi_n = i\right), \quad \sum_{j \neq i} p_{ij} = 1.$$

Let us introduce the conditional distribution function $F_{ij}(t)$ of duration of residence in i before transition to j provided that namely this transition is realized. By definition,

$$F_{ij}(t) = P\left(t_{ij} < t \,|\, \xi(0) = i, \xi(t) = j\right).$$

Then

$$p_{ij}(t) = F_{ij}(t) \cdot p_{ij}. \tag{4}$$

The matrix of transition probabilities of the embedded Markov chain P=(p_{ij}) together with the matrix of conditional functions of residence time distribution, $F(t) = \left(F_{ij}(t)\right)$, i, $j \in E$, and the initial state determine the third way of specifying the semi-Markov process.

Define the unconditional function of distribution of residence time i before leaving for some other state

$$F_i(t) = P\left(t_i < t\right) = \sum_{j \in E, \, j \neq i} p_{ij} F_{ij}(t) = \sum_{j \in E, \, j \neq i} p_{ij}(t). \tag{5}$$

The same function can be defined in a different way, through the original matrix $\left(Q_{ij}(t)\right)$:

$$F_i(t) = 1 - \prod_{i \in E, \, j \neq i} \left(1 - Q_{ij}(t)\right). \tag{6}$$

The expression corresponding to (5) for the density of distribution of the residence time in i has the form

$$f_i(t) = \sum_{j \in E, \, j \neq i} p_{ij} \frac{dF_{ij}(t)}{dt} = \sum_{j \in E, \, j \neq i} p_{ij} f_{ij}(t). \tag{7}$$

Introduce additionally the matrix of conditional transition probabilities:

$$q(t) = (q_{ij}(t)), \ i, \ j \in E,$$

$$q(t) = P(\xi(t) = j \,|\, t_{ij} = t, \ \xi(U) = i).$$

Thus, $q_{ij}(t)$ is the conditional probability that the SMP $\xi(t)$ falls into the state j after being in the state i for a time equal to t. It is clear that

$$p_{ij} = P(\xi(t) = j, \ t_{ij} < t \,|\, \xi(0) = i) = \int_0^t q_{ij}(\tau) f_i(\tau) \, d\tau.$$

The matrix of conditional transition probabilities $\left(q_{ij}(t)\right)$, the vector of the functions of distribution of the residence time $\left(F_i(t)\right)$, $i \in E$, and the initial state determine the fourth way of specifying the SMP.

All above methods of specifying SMP are absolutely equivalent. They equally allow one to analyze behavior of the system described by the semi-Markov model. However, they are far from being equal in terms of the amount of information necessary to obtain a relevant source data. As already mentioned, it is most simple in this case to obtain a matrix of independent distribution functions $Q_{ij}(t)$, i, $j \in E$.

Introduce the functions

$$\psi_i(t) = 1 - F_i(t) = \int_t^\infty f_i(t) \, dt = P\left(t_i > t\right), \quad i \in E. \tag{8}$$

The function $\psi_i(t)$ determines the probability that the process that was in the state i at the zero time will have no time to leave this state in time t.

Let $\Phi_{ij}(t)$ be the conditional probability that the system is in the state j at the time instant t if it was in the state i at the time instant t=0. This probability is called interval-transient probability.

The system starting from i, can enter the state j at the time instant t in different ways.

First, if i=j, then it may not leave i for the entire time t or when leaving i, at least once, it will return to i on the time

instant t. The corresponding interval-transient probability is determined by correlation

$$\Phi_{ij}(t)=\psi_i(t)+\sum_{k\in E,\,k\neq i}p_{ik}\int_0^t f_{ik}(\tau)\cdot\Phi_{ki}(t-\tau)\mathrm{d}\tau. \quad (9)$$

Secondly, if $j\neq i$, then the system can get into this state j taking an intermediate state k at some time instant $\tau<t$ while

$$\Phi_{ij}(t)=\sum_{k\in E,\,k\neq i}p_{ik}\int_0^t f_{ik}(\tau)\cdot\Phi_{kj}(t-\tau)\mathrm{d}\tau,\quad j\neq i. \quad (10)$$

Combine (9) and (10) to get:

$$\Phi_{ij}(t)=\delta_{ij}\psi_i(t)+\sum_{k\in E,\,k\neq i}p_{ik}\int_0^t f_{ik}(\tau)\cdot\Phi_{kj}(t-\tau)\mathrm{d}\tau,\ i,\ j\in E. \quad (11)$$

Here,

$$\delta_{ij}=\begin{cases}1, i=j,\\ 0, i\neq j.\end{cases}$$

The system of linear integral equations (11) allows us to find interval-transient probabilities. Solution for the system can be obtained using the Laplace transform.

Let

$$\Phi^*_{ij}(s)=\delta_{ij}\psi^*_i(s)+\sum_{k\in E,\,k\neq i}p_{ik}\int f^*_{ik}(s)\cdot\Phi^*_{kj}(s),\ i,\ j\in E. \quad (12)$$

Further, apply the Laplace transform and obtain

$$\psi^*_i(s)=\int_0^\infty e^{-st}\left[1-\int_0^t f_i(\tau)\mathrm{d}\tau\right]\mathrm{d}t=$$
$$=\int_0^\infty e^{-st}\mathrm{d}t-\int_0^\infty e^{-st}\left(\int_0^t f_i(\tau)\mathrm{d}\tau\right)\mathrm{d}t=$$
$$=-\frac{e^{-st}}{s}\Bigg|_0^\infty-\frac{f^*_i(s)}{s}=\frac{1}{s}\left(1-f^*_i(s)\right). \quad (13)$$

Write now the system (12) in a matrix form. To do this, introduce matrices

$$\Phi(t)=(\Phi_{ij}(t)),\ \ P=\left(p_{ij}\right);\ \ f(t)=\left(f_{ij}(t)\right),$$

diagonal matrices

$$W(t)=\left(\delta_{ij}f_i(t)\right),\ \ F_i(t)=\left(\delta_{ij}F_i(t)\right),\ \ \psi(t)=\left(\delta_{ij}\psi_i(t)\right)$$

and corresponding Laplace transforms. Besides, to take into account the specific form of summation in (12), introduce a special matrix multiplication operation denoting it by symbol ○. In accordance with this operation, for square matrices of the same dimensionality A, B and C, the notation C=A○B means that $c_{ij}=a_{ij}\cdot b_{ij}$, $i,\ j\in E$.

Then

$$\Phi^*(s)=\psi^*(s)+\left[P\circ f^*(s)\right]\Phi^*(s), \quad (14)$$

whence

$$\Phi^*(s)=\left[1-P\circ f^*(s)\right]\psi^*(s). \quad (15)$$

It follows from (15) that the interval-transient probabilities depend only on the products p_{ij} and $f_{ij}(t)$ and not on each of the factors separately. Using (15), one can investigate the transient process in the system and the stationary distribution of the state probabilities. Taking into account (13),

$$\psi(s)=\frac{1}{s}(1-W).$$

Then

$$\Phi^*(s)=\left[1-P\circ f^*(s)\right]^{-1}\frac{1}{s}(1-W).$$

5. Application of the technology of probability-time graphs

Complexity of analytical solution of the system of equations (12) depends on the nature of the $f(t)$ functions describing density of the distribution of duration of the system residence in each of its possible states before leaving. The following problem seriously complicates the possibility of real use of semi-Markov methods for solving practical problems: the fundamental necessity of knowing densities of distribution of the residence time in each of the states and the difficulties of their adequate correct description for real systems. At the same time, in many cases, the system study objectives are simplified to find the probabilities of achieving a certain final state from the initial and average time to reach this state. This problem is solved using the mathematical apparatus of probability-time graphs [13]. The technology of probability-time graphs was developed to calculate statistical characteristics of routes in computer networks between the given initial and final points. In this case, the set of intermediate points is displayed by an oriented graph, the arcs of which specify a set of possible transitions. Each arc is attributed with a probability P_i of transition along this arc and the transition time T_i. This method is based of the technology of forming a certain function designed in a definite way:

$$f_i(P_i,T_i)=P_iz^{T_i},\ \ i=1,2,..,n \quad (16)$$

called the generating function.

When using (16) for some route of i_1, i_2, ... i_l transitions, an equivalent generating function can be obtained.

$$F_{i_1,i_2,..,i_l}(z)=\prod_{k=1}^{l}P_kz^{T_k}=\left(\prod_{k=1}^{e}P_k\right)Z^{\sum_{k=1}^{e}T_k}. \quad (17)$$

However, if the transition from some initial state to the final one is realized by a system of parallel arcs $i_1, i_2, ... i_l$, the corresponding equivalent generating function will have the form

$$F_{i_1,i_2,..,i_l}(z)=\sum_{k=1}^{l}P_kz^{T_k}. \quad (18)$$

Besides, loops may appear on any route during the system operation. For example, if an error is detected during testing, it is necessary to eliminate it and re-test. Suppose, for example, that for an arc joining vertices i_1 and i_2 with a generating function $f_{i_1}(z)$, a loop appears that connects vertices i_2 and i_1 with a generating function $f_{i_2}(z)$. In this case, taking into

account possible multiple returns from i_2 to i_1, the equivalent generating function will be described by the correlation

$$F_{i_1 i_2}(z)=f_{i_1}(z)+f_{i_1}(z)(f_{i_1}(z)f_{i_2}(z))+$$
$$+f_{i_1}(z)(f_{i_1}(z)f_{i_2}(z))^2+\ldots=$$
$$=f_{i_1}(z)[1+\sum_{k=1}^{\infty}(f_{i_1}(z)f_{i_2}(z))^k]=$$
$$=f_{i_1}(Z)\left[1+\frac{f_{i_1}(Z)f_{i_2}(Z)}{1-f_{i_1}(Z)f_{i_2}(Z)}\right]=$$
$$=\frac{f_{i_1}(z)}{1-f_{i_1}(z)f_{i_2}(z)}=\frac{P_{i_1}z^{T_{i_1}}}{1-P_{i_1}P_{i_2}z^{T_{i_1}+T_{i_2}}}. \quad (19)$$

The choice of correlation for description of the generating function (16) is made in such a way that numerical characteristics of the random path transit time could be obtained when z=1. In this case, the mean transit time is calculated by formula

$$\overline{T}_{i_1,i_2,\ldots i_l}=\left.\frac{dF_{i_1 i_2\ldots i_l}(z)}{dz}\right|_{z=1}, \quad (20)$$

and the variance of the random time is determined in accordance with the correlation

$$D[T_{i_1 i_2\ldots i_l}]=$$
$$=\left[\frac{d^2F_{i_1,i_2,\ldots i_l}(z)}{dz^2}+\frac{dF_{i_1,i_2,\ldots i_l}(z)}{dz}-\left(\frac{dF_{i_1,i_2,\ldots i_l}(z)}{dz}\right)^2\right]_{z=1}. \quad (21)$$

The use of (20) to calculate the mean time for a sequential transition along the route i_1, i_2, ... i_l gives

$$\overline{T}_{i_1,i_2,\ldots i_l}=\left.\frac{d\left(\prod_{k=1}^{l}P_k z^{T_{i_1}+T_{i_2}+\ldots+T_{il}}\right)}{dz}\right|_{z=1}=\left(\prod_{k=1}^{l}P_{i_k}\right)\cdot\left(\sum_{k=1}^{l}T_{i_k}\right), \quad (22)$$

and in transition along a system of parallel arcs, the following is obtained

$$\overline{T}_{i_1,i_2,\ldots i_l}=\left.\frac{d\left(\sum_{k=1}^{l}P_{i_k}z^{T_k}\right)}{dz}\right|_{z=1}=\sum_{k=1}^{l}P_{i_k}T_{i_k}. \quad (23)$$

Finally, the mean time to overcome the loop between nodes i_1, i_2 is calculated taking into consideration (19):

$$\overline{T}_{i_1,i_2}=\left.\frac{d\left(P_{i_1}z^{T_{i_1}}\Big/\left(1-P_{i_1}P_{i_2}z^{T_{i_1}+T_{i_2}}\right)\right)}{dz}\right|_{z=1}=$$
$$=\left.\frac{P_{i_1}T_{i_1}z^{T_{i_1}-1}\left(1-P_{i_1}P_{i_2}z^{T_{i_1}+T_{i_2}}\right)+P_{i_1}P_{i_2}\left(T_{i_1}+T_{i_2}\right)z^{T_{i_1}+T_{i_2}-1}P_{i_1}z^{T_{i_1}}}{\left(1-P_{i_1}P_{i_2}z^{T_{i_1}+T_{i_2}}\right)^2}\right|_{z=1}=$$
$$=\frac{P_{i_1}T_{i_1}(1-P_{i_1}P_{i_2})+P_{i_1}{}^2P_{i_2}\left(T_{i_1}+T_{i_2}\right)}{\left(1-P_{i_1}P_{i_2}\right)^2}=\frac{P_{i_1}T_{i_1}-P_{i_1}{}^2P_{i_2}T_{i_1}+P_{i_1}{}^2P_{i_2}T_{i_1}+P_{i_1}{}^2P_{i_2}T_{i_2}}{\left(1-P_{i_1}P_{i_2}\right)^2}=\frac{P_{i_1}T_{i_1}+P_{i_1}{}^2P_{i_2}T_{i_2}}{\left(1-P_{i_1}P_{i_2}\right)^2}. \quad (24)$$

The use of probability-time graphs makes it possible to simplify topological description of complex objects by constructing equivalent graphs.

Fig. 1 presents a generalized probability-time graph of the software security testing algorithm.

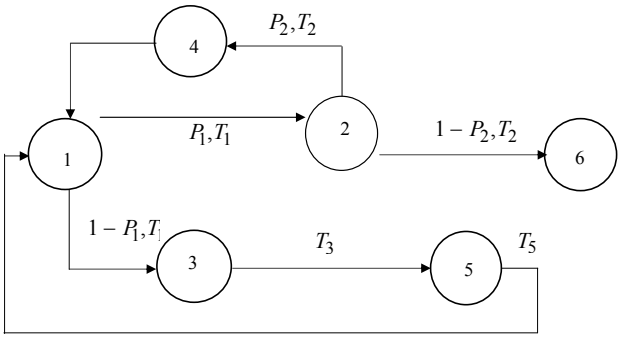

Fig. 1. Generalized model of the testing algorithm

In this model, vertex 1 corresponds to a match check operation, vertex 2 corresponds to the operation of evaluating the check completeness. At the node corresponding to vertex 3, criticality of mismatch is estimated. At the node 4, incompleteness level is evaluated. At the node 5, errors are corrected. The node 6 determines success of total check. The graph arcs define transitions between the corresponding vertices. All above information on the specification of the graph elements is given in Table 12 together with data on transition probabilities and duration of operations.

Table 1

Characteristics of the vertices of the probability-time graph

Item No.	Vertex number	Operation name	Operation duration
1	1	Match check	T_1
2	2	Completeness check	T_2
3	3	Mismatch criticality estimation	T_3
4	4	Incompleteness level estimation	T_4
5	5	Error correction	T_5

Table 2

Characteristics of the arcs of the probability-time graph

Item No.	Arc number	Operation name	Transition probability
1	1–2	Transition to the completeness check	P_1
2	1–3	Transition to the mismatch criticality check	$1-P_1$
3	2–4	Transition to the completeness level check	P_2
4	2–6	Transition to the test completion	$1-P_2$
5	3–5	Transition to the error correction	1
6	4–1	Return to the initial state	1
7	5–1	Return to the initial state	1

Let us analyze the probability-time graph.

Using the relations introduced, simplify the graph by "gluing" together the vertices 3 and 5. In accordance with

(17), the equivalent generating function of this graph branch will have the form:

$$F_{1351}(Z)=(1-P_1)Z^{T_3+T_5}.$$

Further, using (19), introduce the generating function for the loop $(i_1\ i_2\ i_4\ i_1)$.

$$F_{1241}(Z)=\frac{P_1Z^{T_1}}{1-P_1P_2Z^{T_2+T_4}}.$$

Now, the initial probability-time graph is simplified to the form shown in Fig. 2.

In this case, the vertex $\tilde{2}$ and the vertex $\tilde{3}$ are the results of "gluing" the vertices (2, 4) and (3, 5), respectively. The average duration of overcoming the loops (1, 2, 4, 1) and (1, 3, 5, 1) in accordance with (24) are determined by correlations:

$$\tilde{T}_2=\frac{P_1T_1+P_1^2(T_2+T_4)}{(1-P_1P_2)^2},\ \ \tilde{T}_3=\frac{(1-P_1)T_1+(1-P_1)^2(T_3+T_5)}{P_1^2}.$$

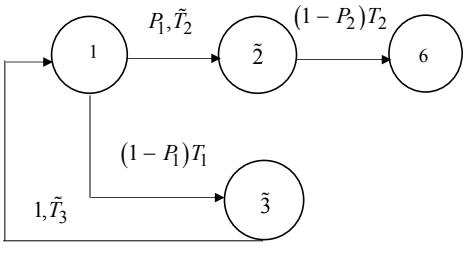

Fig. 2. The simplified model of the testing algorithm

Next, calculate the average time of transit from the initial node 1 to the final node 6:

$$\tilde{T}_{16}=T_1+\frac{P_1T_1+P_1^2P_2(T_2+T_4)}{(1-P_1P_2)^2}.$$

Finally, taking into account the possible time spent on elimination of the mismatch errors, the average test duration will be

$$T=P_1\tilde{T}_2+(1-P_1)\tilde{T}_3=$$
$$=\frac{P_1^2T_1+P_2(T_2+T_4)}{(1-P_1P_2)^2}+\frac{(1-P_1)^2T_1+(1-P_1)^3(T_3+T_5)}{P_1^2}.$$

The above correlations allow us to determine the law of distribution of the random duration of the testing procedure. To this end, compile a table of events that occur during implementation of this procedure with indication of their probabilities and the time of their occurrence elapsed from the start of testing (Table 3).

These events are incompatible and in aggregate, they form a complete group. Therefore, the sum of their probabilities should be equal to 1.

Determine the sum Σ_1 of probabilities of the events corresponding to the completion of testing, the sum Σ_2 of probabilities of the events corresponding to the detection of a mismatch, and the sum Σ_3 of probabilities of the events corresponding to the mismatch elimination and obtain the following:

$$\sum\nolimits_1=\sum_{k=0}^{\infty}P_1\left[(P_1P_2)^k\right](1-P_2)=\frac{P_1(1-P_2)}{1-P_1P_2},$$

$$\sum\nolimits_2=\sum_{k=0}^{\infty}P_1\left[(P_1P_2)^k\right]P_2(1-P_1)=\frac{P_1P_2(1-P_1)}{1-P_1P_2},$$

$$\sum\nolimits_3=\sum_{k=1}^{\infty}(1-P_1)^k\,P_1=P_1\cdot\frac{1-P_1}{1-(1-P_1)}=1-P_1,$$

while

$$\sum\nolimits_1+\sum\nolimits_2+\sum\nolimits_3=\frac{P_1(1-P_2)}{1-P_1P_2}+\frac{P_1P_2(1-P_1)}{1-P_1P_2}+1-P_1=$$
$$=\frac{P_1(1-P_2)+P_1P_2(1-P_1)}{1-P_1P_2}+1-P_1=$$
$$=P_1\frac{1-P_2+P_2-P_1P_2}{1-P_1P_2}+1-P_1=1.$$

Table 3

A fragment of the list of events in the testing process. Probability-time characteristics

Item No.	Event name	Probability	Time from the process start
1	Completion of the unproblematic testing procedure	$P_1(1-P_2)$	T_1+T_2
2	Detection of incompleteness. Transition to the node 4	P_1P_2	T_1+T_2
3	Detection of mismatch after a single cycle of incompleteness check	$P_1\left[P_2(1-P_1)\right]$	$2T_1+\tilde{T}_2$
4	Completion of testing after a single cycle of completeness check	$P_1(P_2P_1)(1-P_2)$	$2T_1+2\tilde{T}_2$
5	Completion of testing after a double cycle of completeness check	$P_1\left[(P_2P_1)^2\right](1-P_2)$	$3T_1+2\tilde{T}_2+T_2$
...	...	...	...
k	Completion of testing after the k-th cycle of incompleteness check	$P_1\left[(P_2P_1)^k\right](1-P_2)$	$(k+1)T_1+k\tilde{T}_2+T_2$
$k+1$	Detection of mismatch after the k-th cycle of incompleteness check	$P_1\left[(P_2P_1)^k\right]P_2(1-P_1)$	$(k+2)T_1+k\tilde{T}_2+T_2$

Thus, the law of distribution of the test duration is obtained. It can be used in working out measures to improve effectiveness of the testing procedure.

6. Discussion of results of studying the developed mathematical models of the testing algorithm

In this paper, main lines from the arsenal of methods of the theory of complex systems were considered. The use of these methods enables construction of adequate mathe-

matical models of real systems of a complicated structure characteristic of a software testing algorithm.

Let us consider advantages and disadvantages of the proposed models.

The models of the first type realize known methods of the theory of semi-Markov processes. It was shown that this technology for calculation of statistical characteristics of systems while being irreproachable in stringency of its construction, solves the problem of evaluating quality of testing algorithms. However, accuracy of the results obtained in this case essentially depends on accuracy of description of the laws of distribution of residence time in each system state.

The alternative approach consists in the use of probability-time graphs and is much less demanding. For its implementation, it is sufficient to know the mean time of residence in each state and the probability of transitions from one state to another. This approach was at design steps and brought to the process of calculating the law of distribution of the testing procedure execution time. It should be noted that the mathematical model of the testing process obtained by the method of probability-time graphs is informative and useful *per se*, even in absence of numerical values of initial data. This model determines the explicit dependence of the performance indicators of the testing algorithm on the values of the statistical characteristics of the algorithm which is available for direct analysis.

The obtained theoretical relationships can really be used to develop preliminary recommendations and possible ways of improving effectiveness of the software testing algorithms.

Possible lines for further studies are connected with the necessity of taking into account the real uncertainty of the source data which can be described fuzzily [15–18] or incorrectly [19]. The problems arising in this case are solved based on the methods proposed in [20, 21].

7. Conclusions

1. A graphic-analytical mathematical model of the CSCA software security testing algorithm has been developed. It differs from known models by its taking into account specifics of software security testing in the process of mathematical formalization. The model can be used to study basic stages of the CSCA software security testing. Application of this model will reduce the software vulnerability and increase security of the IT project in general. Also, the model is applicable when developing new methods, algorithms, and procedures for managing the IT projects.

2. Application of the method of probability-time graphs in the course of mathematical modeling enables use of the results obtained in an analytical form. These results (distribution density functions) are used in conducting comparative analyses and studies of various stages and steps of software testing.

References

1. Tobias K. A Bug Hunter's Diary: A Guided Tour Through the Wilds of Software Security. No Starch Press, 2011. 208 p.
2. The Software Improvement Group (SIG) measures software energy use. The Green IT Review. URL: http://www.seflab.com/news/the-software-improvement-group-sig-measures-software-energy-use-the-green-it-review/
3. Resource-Oriented Approaches to Implementation of Traffic Control Technologies in Safety-Critical I&C Systems / Kuchuk G., Kovalenko A., Kharchenko V., Shamraev A. // Studies in Systems, Decision and Control. 2017. P. 313–337. doi: 10.1007/978-3-319-55595-9_15
4. Semenov S. H., Kassem Khalife Kompleks matematychnykh modelei protsessu rozrobky prohrammnoho zabezpechennia // Informatsiyni tekhnolohiyi ta kompiuterna inzheneriya. 2017. Issue 3 (40). P. 61–68.
5. Green IT Engineering: Concepts, Models, Complex Systems Architectures / V. Kharchenko, Y. Kondratenko, J. Kacprzyk (Eds.) // Studies in Systems, Decision and Control. 2017. doi: 10.1007/978-3-319-44162-7
6. Annadurai C. Review of Packet Scheduling Algorithms in Mobile Ad Hoc Networks // International Journal of Computer Applications. 2011. Vol. 15, Issue 1. P. 7–10. doi: 10.5120/1914-2552
7. Bordunov I. B., Kosachev A. S. Polnoe testirovanie s otkrytym sostoyaniem ogranichenno nedeterminirovannyh sistem // Programmirovanie. 2009. Issue 6. P. 3–18.
8. Multiservice network security metric / Mozhaev O., Kuchuk H., Kuchuk N., Mozhaev M., Lohvynenko M. // 2017 2nd International Conference on Advanced Information and Communication Technologies (AICT). 2017. doi: 10.1109/aiact.2017.8020083
9. Burdonov I. B., Kosachev I. S. Bezopasnoe testirovanie simulyacii sistem s otkazami i razrusheniem // Modelirovanie i analiz informacionnyh sistem. 2010. Vol. 17, Issue 4. P. 27–40.
10. Goodmon L. B., Anderson M. C. Semantic integration as a boundary condition on inhibitory processes in episodic retrieval // Journal of Experimental Psychology: Learning, Memory, and Cognition. 2011. Vol. 37, Issue 2. P. 416–436. doi: 10.1037/a0021963
11. Approaches to selection of combinatorial algorithm for optimization in network traffic control of safety-critical systems / Kuchuk G., Kharchenko V., Kovalenko A., Ruchkov E. // 2016 IEEE East-West Design & Test Symposium (EWDTS). 2016. doi: 10.1109/ewdts.2016.7807655
12. Krishnan M. S. Software Development Risk Aspects and Success Frequency on Spiral and Agile Model // International Journal of Innovative Research in Computer and Communication Engineering. 2015. Vol. 03, Issue 01. P. 301–310. doi: 10.15680/ijircce.2015.0301024
13. Raskin L. G. Matematicheskie metody issledovaniya operaciy i analiza slozhnyh sistem vooruzheniya PVO. Kharkiv: VIRTA PVO, 1988. 178 p.
14. Zhuravlev A. Yu. Polumarkovskie processy. Moscow, 2014. 218 p.

15. Bellman R. E., Zadeh L. A. Decision-Making in a Fuzzy Environment // Management Science. 1970. Vol. 17, Issue 4. P. B-141–B-164. doi: 10.1287/mnsc.17.4.b141
16. Raskin L. G., Seraya O. V. Nechetkaya matematika. Kharkiv: Parus, 2008. 352 p.
17. Kaufman A., Gupta M. Introduction to Fuzzy Arithmetic: Theory and Applications. New York: VN. Reinhold, 1985. 351 p.
18. Liu B., Liu Y.-K. Expected value of fuzzy variable and fuzzy expected value models // IEEE Transactions on Fuzzy Systems. 2002. Vol. 10, Issue 4. P. 445–450. doi: 10.1109/tfuzz.2002.800692
19. Pawlak Z. Rough sets // International Journal of Computer & Information Sciences. 1982. Vol. 11, Issue 5. P. 341–356. doi: 10.1007/bf01001956
20. Raskin L., Sira O. Fuzzy models of rough mathematics // Eastern-European Journal of Enterprise Technologies. 2016. Vol. 6, Issue 4 (84). P. 53–60. doi: 10.15587/1729-4061.2016.86739
21. Raskin L., Sira O. Method of solving fuzzy problems of mathematical programming // Eastern-European Journal of Enterprise Technologies. 2016. Vol. 5, Issue 4 (83). P. 23–28. doi: 10.15587/1729-4061.2016.81292

Досліджуються кусково-поліноміальні криві третього степеня. Вводиться послідовність точок, які розглядаються як керуючі, а з'єднуючі їх відрізки є дотичними до кривої. Побудовано систему рівнянь для обчислення коефіцієнтів кривої та знайдено умови її єдиності. На прикладах розрахунків показано хороші апроксимаційні властивості одержаної кривої та проілюстрована можливість локальної зміни її форми в залежності від параметрів

Ключові слова: сплайнова крива третього степеня, крива Без'є, параметри форми кривої

Исследуются кусочно-полиномиальные кривые третьей степени. Вводится последовательность точек, рассматриваемых как управляющие, а соединяющие их отрезки являются касательными к кривой. Построена система уравнений для вычисления коэффициентов кривой и найдены условия ее единственности. На примерах расчетов показаны хорошие аппроксимационные свойства полученной кривой и проиллюстрирована возможность локального изменения ее формы в зависимости от параметров

Ключевые слова: сплайновая кривая третьей степени, кривая Безье, параметры формы кривой

UDC 519.6:517.51
DOI: 10.15587/1729-4061.2018.128284

APPLICATION OF PIECEWISE-CUBIC FUNCTIONS FOR CONSTRUCTING A BEZIER TYPE CURVE OF C^1 SMOOTHNESS

O. Stelia
PhD, Associate Professor, Senior Researcher*
E-mail: oleg.stelya@gmail.com
L. Potapenko
PhD*
E-mail: lpotapenko@ukr.net
I. Sirenko*
E-mail: i.sirenko@gmail.com
*Laboratory of the Computational Methods in the Mechanics of Continuous Media
Taras Shevchenko National University of Kyiv
Volodymyrska str., 60, Kyiv, Ukraine, 01033

1. Introduction

Interpolation and approximation of numerical sets of data is a relevant task in applied mathematics because of a widespread application in various fields of science and technology. Among the various areas of research, there are two that stand out – the interpolation using polynomial splines, which solves a problem on the construction of curves that pass through the set points, and methods for constructing Bezier curves for which set points are the control points. It is understood in the sense that a curve does not pass through these points but approaches them, changing in so doing its shape depending on their location. At present, researchers chose to combine these two approaches. That makes it possible to obtain rather smooth curves and efficient algorithms for their construction with the possibility of interactive control over the shape of the curves using control points.

2. Literature review and problem statement

A method for constructing curves, which are called the Bezier curves, was developed independently by engineers Pierre Bézier, who worked for the automotive company Renault (Headquarters in the city of Boulogne-Billancourt, France), and Paul de Castillo, who was an employee of the automobile company Citroën (Headquarters in Paris, France) [1]. They proposed to apply these curves to design automobile bodies. A widespread use of Bezier curves for the problems on approximation is associated with convenience in both the analytical description and the visual geometrical construction. Employing the Bezier curves in computer graphics systems allows the user to move control points using a cursor on the screen to interactively change the shape of the curve [2]. This is a handy tool used in various areas of technical design.