Science in China Series F: Information Sciences

www.scichina.com
info.scichina.com
www.springerlink.com

Complexity of software trustworthiness and its dynamical statistical analysis methods

ZHENG ZhiMing[1,3†], MA ShiLong[2†], LI Wei[2], JIANG Xin[1], WEI Wei[3], MA LiLi[1] & TANG ShaoTing[1]

[1] Key Laboratory of Mathematics, Informatics and Behavioral Semantics, Ministry of Education, Beihang University, Beijing 100191, China;

[2] State Key Laboratory of Software Development Environment, Beihang University, Beijing 100191, China;

[3] School of Mathematical Sciences, Peking University, Beijing 100871, China

Developing trusted softwares has become an important trend and a natural choice in the development of software technology and applications. At present, the method of measurement and assessment of software trustworthiness cannot guarantee safe and reliable operations of software systems completely and effectively. Based on the dynamical system study, this paper interprets the characteristics of behaviors of software systems and the basic scientific problems of software trustworthiness complexity, analyzes the characteristics of complexity of software trustworthiness, and proposes to study the software trustworthiness measurement in terms of the complexity of software trustworthiness. Using the dynamical statistical analysis methods, the paper advances an invariant-measure based assessment method of software trustworthiness by statistical indices, and hereby provides a dynamical criterion for the untrustworthiness of software systems. By an example, the feasibility of the proposed dynamical statistical analysis method in software trustworthiness measurement is demonstrated using numerical simulations and theoretical analysis.

software trustworthiness, trustworthiness attributes, dynamical system, invariant-measure

1 Introduction

As demands on software functions increase continuously, software systems have become increasingly huge and very hard to manage. It is difficult to avoid software bugs and system loopholes. The systems do not work in an expected way very often, and faults and failures arise frequently, which directly or indirectly cause the losses for users. Therefore, software systems are not always trustable[1]. Thus, how to guarantee the software trustworthiness in design, development, operation and maintenance of software systems has become one of the great challenges in software technology study.

The software trustworthiness, as a new concept, integrates related concepts such as correctness, reliability, safety and security, timeliness, completeness, availability, predictability, survivability and controllability, comprehensively reflecting various attributes of objective entities in reality[2−6]. According to the difference in applications, software

Received July 29, 2008; accepted October 23, 2008; published online August 4, 2009
doi: 10.1007/s11432-009-0143-4
†Corresponding authors (email: zzheng@pku.edu.cn, slma@nlsde.buaa.edu.cn)
Supported by the National Basic Research Program of China (Grant No. 2005CB321900), and the National Natural Science Foundation of China (Grant No. 60473091)

Citation: Zheng Z M, Ma S L, Li W, et al. Complexity of software trustworthiness and its dynamical statistical analysis methods. Sci China Ser F-Inf Sci, 2009, 52(9): 1651–1657, doi: 10.1007/s11432-009-0143-4

trustworthiness can be characterized using a set of related trustworthiness attributes. In case some attributes are exceptional in running, then some problems may arise with software trustworthiness. These attributes for software trustworthiness are used for judging whether or not the computing behaviors and the computing results in running of software systems can meet the expectations in dynamical and open environments, and whether the systems can survive, provide functions and guarantee the QoS even under external interferences.

Currently, the scale and complexity of software systems, and the scale and complexity of components composing the systems are increasing. Take embedded software systems as an example. In the software for 2005 Audi A8 the length of the code is over 90 MB, and, in the software for AirBus A380 with 400 MB code length, about 100 functions are provided[7]. The scale and complexity of software systems for Internet are even greater. In the meantime, with the development of computing mode towards network environments and service-oriented architecture, the running environments of software systems, including network environments and physical environments, are more open and are dynamically changing constantly. The wide applications of wireless technology and open source softwares are two examples[8]. They, together with the constant evolution of software systems[9], have brought much more complexity. Under the effect of multiple factors of internal risks, e.g., errors and bugs, and external risks, e.g., viruses and malicious codes, the software systems survive and evolve in dynamical complex environments, and their trustworthiness exhibits procedural, structural and nonlinear complexity, bringing huge challenges to software trustworthiness study. Currently, because the behaviors of softwares are highly complex and difficult to predict quantitatively, trustworthiness is verified by collecting data which are used for finding the evidence in most of trustworthiness measurement methods. However, the data collection depends on human interactions and relative and comprehensive considerations for multiple trustworthiness attributes are lacking[2]. Meanwhile, the current research methods are more technical[10], and they neglect the fact that there may exist laws governing behaviors of software systems that are not governed by human wills.

Software trustworthiness is for the purposes of overall characterizing and measuring the behaviors of software systems, and comprehensively considering the characteristic attributes of reliability, safety and security, fault-tolerance and so on. In this sense, quantitative indices, measurement and assessment mechanisms for software trustworthiness complexity based on the software trustworthiness dynamical models[11] are the key research topics. The approach to software trustworthiness complexity based on dynamics is trying to explore the statistical laws governing ultimate evolutionary behaviors of software systems using theory and methods in dynamical statistical analysis.

In ref. [11], the authors establish dynamical models for software trustworthiness, and demonstrate the relationships between characteristics of software behaviors and dynamical systems. Based on it, by using the theory and methods in dynamical statistical analysis, this paper studies the statistical laws governing ultimate evolutionary behaviors of software systems by multiple trustworthiness attributes, and studies the software trustworthiness complexity with invariant-measure method. For trustworthiness of softwares composed of components, this paper proposes a measurement method with multi-scale trustworthiness indices.

2 Dynamical evolution models for software trustworthiness

Generally, the software system trustworthiness includes identity trustworthiness, function trustworthiness and behavior trustworthiness, in which the behavior trustworthiness, the central problem in software trustworthiness study, involves with the behavior evolutions of software systems. The behavior trustworthiness relates to networks, OS's, middlewares, human-computer interactions and so on. Thus, the behavior trustworthiness of software systems can be considered as that the behaviors and results of software systems can be predicted,

states can be monitored, results can be assessed, and exceptions can be controlled.

Because software systems dynamically evolve in their life cycle in open and complex environments, the software trustworthiness is evolving under the effect of both internal and external factors. Factors affecting the software trustworthiness mainly include internal crisis, i.e., the errors, bugs, faults, aging and failures, etc.[12], external crisis, i.e., virus, malicious codes, etc., and the survival evolutions of software systems in complex and changing environments. According to the factors affecting software trustworthiness mentioned above, combining with the laws of natural evolution, evolution with human interactions and survival evolution of software systems, software trustworthiness evolutions can be modeled as $F : M \times N \to N$, in which M is the space of software trustworthiness attributes, $M \subset R^m$, m the number of the attributes under consideration, N the parameter space characterizing external factors, $N \subset R^n$, and n the number of parameters. A given set of external parameter values determines a specific external environment. Considering the characteristics of internal and external factors, the evolutionary law of the software trustworthiness can be formulated as $F = f \circ g$. Here F takes form of composition of maps, in which f reflects the internal (natural) evolutionary law and g reflects the external (human interaction and survival) evolutionary law. Thus, software trustworthiness evolution can be represented by the iterative behaviors of dynamical system F.

The above software trustworthiness evolution models give the interpretations of software trustworthiness in dynamics. That is, predicting the behaviors and results of software systems corresponds to a class of dynamical behaviors of dynamical systems, monitoring the behavioral states corresponds to the orbit analysis of dynamical systems, assessing the behavioral results corresponds to the behavioral analysis of the limit sets of dynamical systems, and controlling the behavioral exceptions corresponds to the study of structural stability and bifurcation of dynamical systems.

3 Complexity of software trustworthiness and its analysis by statistical indices

For the comprehensive trustworthiness of software systems, by studying the limit states of map F in the dynamical model of software trustworthiness evolution, the measurement of software trustworthiness can be studied by exploring the complexity of software trustworthiness. In the viewpoints of the behavior characteristics in the limit states of evolutions, we consider that whether the behaviors and states of software trustworthiness meet the expectations, relating to complexity analysis of limit sets[13,14], and whether the software systems can survive under external attacks, relating to analysis of geometric structure and its stability[15].

3.1 Characteristics of software trustworthiness complexity

Software systems dynamically evolve in their life cycles in open and complex environments. The dynamical evolutions possess the following complexity characteristics, and these complexity characteristics determine the trustworthiness of software systems comprehensively.

For a scientific understanding of the software trustworthiness complexity, it can be considered that there are three types of characteristics of software trustworthiness complexity as follows.

(1) Structural complexity. Since there may be many factors affecting the software trustworthiness, and these factors even may dynamically increase or decrease, and the relationships among the factors may dynamically change, then software trustworthiness exhibits structural complexity. In addition, dynamical and open environments, frequent and uncertain interferences, extensive interactions and large-scale systems, aggravate the structural complexity of software trustworthiness.

(2) Nonlinear complexity. The factors affecting the software trustworthiness may interact to one another; as a result, software trustworthiness exhibits nonlinear complexity.

(3) Procedural complexity. Software trustworthiness is developing and evolving constantly, and

the behaviors and the results of software systems are uncertain, or even unpredictable, very often, leading to the fact that software trustworthiness exhibits procedural complexity.

3.2 Analysis of software trustworthiness complexity by statistical indices

According to the above discussions, the software trustworthiness complexity can be reduced as follows. That is, structural complexity and nonlinear complexity of software trustworthiness are reduced to the complexity of modeling of software trustworthiness, and the procedural complexity is reduced to the complexity of iteration processes and the limit behaviors of the dynamical systems corresponding to software trustworthiness models. In the following, based on the models for software trustworthiness given in ref. [1], this paper will use the dynamical statistical analysis theory and methods to establish the measurement methods for software trustworthiness.

Generally speaking, it is very difficult to analyze the dynamical behaviors quantitatively in mathematics. Thus, the statistical analysis properties, such as, the Lyapunov exponents[16−18], fractal dimensions[18,19] and entropy[20,21], are often used to characterize the basic characteristics of dynamical systems. According to the above characteristics of software trustworthiness complexity, in this paper, the invariant-measure theory, from the statistical analysis in dynamical systems, is used to characterize the statistical laws governing the software trustworthiness complexity. The basic idea of invariant-measure theory is that, the statistics of the average return frequency of each attribute (index) in software systems can be done by tracing and recording the values of the variables that are trustworthiness attributes in software system evolution processes, and hereby the probability distributions of trustworthiness attributes taking certain values in the limit states of software trustworthiness evolutions can be obtained. The invariant-measure theory can reflect the long term behavior characteristics of software trustworthiness, which correspond to the behavior characteristics of limit sets, then revealing the objective overall laws governing software trustworthiness evolutions, providing effective statistical analysis methods for studying the ultimate states and inherent laws for software trustworthiness. In the following, by an example we demonstrate how to use the above dynamical statistical analysis methods for trustworthiness complexity.

Consider an example that the variation of memory consumption is affected by attacks of viruses in the course of the software system running. Here the total memory is supposed to be with whole probability. Denote by $x_n(\geqslant 0)$ the memory consumption at time n. And it is assumed that the relative amount of memory consumption (i.e., x_{n+1}/x_n) affected by the virus attack is the same at different time and this ratio is denoted by parameter $\alpha(\geqslant 0)$. It is also assumed that viruses compete for memory each other, and the reduction in the amount of memory consumption by competition is proportional to x_n^2, and the ratio is denoted by $\beta(\geqslant 0)$. Then, the amount of memory consumption can evolve as follows:

$$x_{n+1} = x_n(\alpha - \beta x_n).$$

Let

$$y_n = \frac{\beta}{\alpha} x_n.$$

Then

$$y_{n+1} = \alpha y_n(1 - y_n).$$

That is, the evolutionary law of the system memory consumption is $F(x, a) = ax(1 - x)$, which is the famous Logistic map[22].

In this example, we just consider a single index, the memory consumption x, which can measure the capacity against the attacks of viruses of the software system. That is, x is an attribute of software trustworthiness measuring the amount of the system memory consumption.

Assume that the initial memory consumption (the memory leakage) amount is zero, that is, the system is not attacked by viruses at the initial time. And the vth iteration of the system is denoted by

$$\xi_v(a) = F^v(0, a),$$

where a is an external parameter characterizing the attacking capacity of viruses. For any given memory consumption amount $x \in [0, 1]$, do statistics for the times of $\xi_v(a)(v = 0, 1, 2, \cdots, n)$ taking x.

Let $\delta_{\xi_v(a)}(x)$ be counting functions, which are 1 for $\xi_v(a) = x$, otherwise, 0. Then, $\sum_{v=1}^{n} \delta_{\xi_v(a)}(x)$ is the number of times of the value of memory consumption amount taking x after n times measurements.

Let

$$\mu_n(x; a) = \frac{1}{n} \sum_{v=1}^{n} \delta_{\xi_v(a)}(x).$$

Then $\mu_n(x; a)$ will weakly converge to $\mu(x; a)$ when $n \to \infty$[23].

Here $\mu_n(x; a)$ is the frequency at which the memory consumption amount takes value x after measuring n times. For any given x, the weak limit $\mu(x; a)$ is the statistical distribution of probability that the memory consumption amount takes the value x, and reflects the objective law of software trustworthiness complexity.

In the above example, for any given subinterval $A \subseteq [0, 1]$, we have

$$\mu(A; a) = \int_A d\mu(x; a),$$

which describes the probability that the memory consumption amount values are in subinterval A.

In the following, we consider an example in which a takes value 4. Then the density function is $d\mu(x; a)/dx = 1/(\pi\sqrt{x(1-x)})$[24,25]. The numerical simulations with $a = 4$ are given in Figure 1.

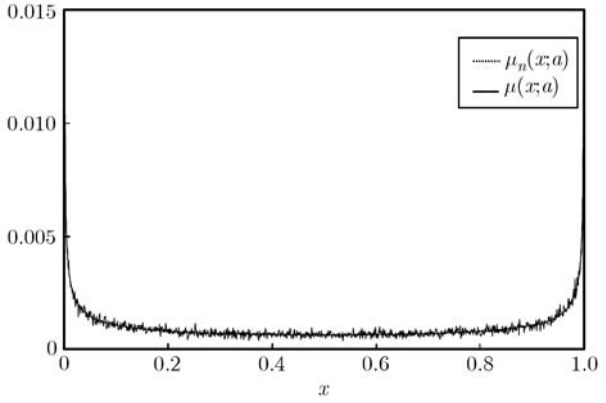

Figure 1 Frequency $\mu_n(x; a)$ versus x with $a = 4$ and $n = 5 \times 10^4$. The dotted curve is the simulation result and solid curve is the theoretical result of $\mu(x; a)$.

Figure 1 demonstrates that $\xi_v(4)(v = 1, 2, \cdots)$ is ergodic in interval [0, 1], in which the memory consumption amount takes values; that is, for any given arbitrarily small interval $A \subseteq [0, 1]$, $\xi_v(4)(v = 1, 2, \cdots)$ can take values in A with a positive probability. In other words, memory consumption amount can certainly take almost all the values in interval [0, 1] in the course of the software trustworthiness evolution, and the probability that the memory consumption amount takes values in any given small intervals exists, thus ensuring that the software trustworthiness can be measurable. This shows that the memory behaviors of software systems exhibit strong randomness under the effect of virus, and the memory behaviors are totally unpredictable after long term evolutions. Consequently, the behaviors of the software system are untrusted when the parameter a characterizing viruses takes value 4.

According to the above example, for a software system F only characterized by a single trustworthiness attribute x, the weak limit $\mu(x; a)$ can be considered as the statistical measurement of the software trustworthiness complexity.

Consider software systems with multiple trustworthiness attributes. Similarly, we can define the weak limit μ_F of

$$\frac{1}{n^m} \left[\sum_{v_1=1}^{n} \sum_{v_2=1}^{n} \cdots \sum_{v_m=1}^{n} \delta_{F^{(v_1, v_2, \cdots, v_m)}(a)}(x) \right]$$

to be the statistical measurement of the software trustworthiness complexity for the software systems. μ_F characterizes the comprehensive software trustworthiness for multiple trustworthiness attributes statistically. If μ_F is invariant and is absolutely continuous under Lebesgue measure (L^1), then the system is ergodic, and the corresponding dynamical behaviors are chaotic, which shows that the behaviors of the corresponding software system are untrusted.

Thus, based on the invariant measure theory of dynamical systems, this paper gives a dynamical criterion for untrustworthiness of software systems, which, in other words, is a necessary condition for software trustworthiness. That is, the behavior characteristics, reflected by the invariant measure, of the software trustworthiness limit evolutions describe the dynamical criterion for software untrustworthiness, i.e., once the mean tendency of the indices of trustworthiness attributes weakly converges to an absolutely continuous invariant

measure, then the dynamical behaviors of software trustworthiness model will certainly evolve to chaos, showing that the behaviors of the corresponding software systems are untrusted eventually.

3.3 Analysis methods of relationships between trustworthiness attributes for softwares composed of atomic components

In distributed computing, software systems are required to share crossing space, time, devices, and users, leading to a great increase in scales, complexity and functions of software systems, promoting the development of software systems towards heterogeneous coordination, integrations, reusability and industry. In order to meet these demands, the new software development paradigm should support distributed computing, modularity, component integrations and so on, making it possible that the software systems can be composed of different atomic components similarly to hardware systems. A software system S can be composed of different atomic components connecting organically, denoted by $S = S(E_1, E_2, \ldots, E_n)$, where $E_i, i = 1, 2, \ldots, n$ are different atomic components composing S. Here, each atomic component E_i possesses trustworthiness attributes. Mathematically, E_i is spanned by the scaling manifold bases of the trustworthiness attributes. According to this basic idea, we can define a scaling function $\Psi = (\Psi_{ij}), i, j = 1, \cdots, n$, where $\Psi_{ij} : E_i \to E_j$ is a conversion map for scale measurements between E_i and E_j. Then, by $\Psi = \Psi(S)$, the measurement function is given to unify the scale measurement of the trustworthiness of software system S. For multi-scale software systems with multiple dimensional attributes, the trustworthiness evolution models can be denoted by $F(\Psi(S)(x), \alpha)$, where F is the dynamical model of the software trustworthiness of system S, and α is the influence parameter of the external environments. Similarly, by using the invariant-measure method to analyze the dynamical behaviors and the complexity of software trustworthiness, we can get the complexity characteristics and the correlation degree among the attributes scaling manifolds of software components.

4 Conclusion

Software trustworthiness is assessed by comprehensively considering the statistical laws governing the software systems with trustworthiness attributes in the course of their evolutions. Based on the dynamical model of software trustworthiness, this paper interprets the characteristics of behaviors of software systems and the basic scientific problems of software trustworthiness complexity, analyzes the characteristics of complexity of software trustworthiness, and indicates that the software trustworthiness measurement can be studied by means of the complexity of software trustworthiness. Using the dynamical statistical analysis methods, the paper proposes an invariant-measure based assessment method of software trustworthiness by statistical indices, and hereby provides a dynamical criterion for the untrustworthiness of software systems. By an example, the feasibility of the proposed dynamical statistical analysis method in software trustworthiness measurement is demonstrated using numerical simulations and theoretical analysis.

1 The Open Trusted Computing (OpenTC) consortium, General activities of OpenTC [EB/OL]. [2006-3-1]. http://www.opentc.net/activities/

2 Liu K, Shan Z G, Wang J, et al. Overview on major research plan of trustworthy software (in Chinese). Bull Nation Nat Sci Found China, 2008, 22(3): 145–151

3 Reith M, Niu J, Winsborough W H. Engineering trusted management into software models. In: Proceedings of the International Workshop on Modeling in Software Engineering. Washington DC: IEEE CS Press, 2007. 9–15

4 Littlewood B, Strigani L. Software reliability and dependability: a roadmap. In: Proceeedings of the Conference on the Future of Software Engineering. New York: ACM Press, 2000. 175–188

5 Banerjee S, Mattmann C, Medvidovic N, et al. Leveraging architectural models to inject trust into software systems. In: Proceedings of the Workshop on Software Engineering for Secure Systems-Building Trustworthy Applications. New York: ACM Press, 2005. 1–7

6 Chen H W, Wang J, Dong W. High confidence software engineering technologies (in Chinese). Acta Elec Sin, 2003, 31(12A): 1933–1938

7 Tiwari V, Malik S, Wolfe A. Power analysis of embedded software: a first step towards softwarepower minimization. IEEE Trans VLSI Syst, 1994, 2(4): 437–445
8 Chatschik B. An overview of the bluetooth wireless technology. IEEE Comm Mag, 2001, 39(12): 86–94
9 Mens T, Demeyer S. Future trends in software evolution metrics. In: Proceedings of the 4th International Workshop on Principles. Austria: ACM Press, 2001. 83–86
10 Tsai W T, Chen Y, Paul R, et al. Cooperative and group testing in verification of dynamic composite Web services. In: Computer Software and Applications Conference. Scotland: IEEE Press, 2004, 2: 170–173
11 Zheng Z M, Ma S L, Li W, et al. Dynamical characteristics of software trustworthiness and their evolutionary complexity. Sci China Ser F-Inf Sci, 2009, 52(8): 1328–1334
12 Huang G, Wang Q X, Mei H, et al. Research on architecture-based reflective middleware (in Chinese). J Software, 2003, 14(11): 1819–1826
13 Mischaikow K, Mrozek M. Chaos in the Lorenz equations: a computer-assisted proof. Bull Amer Math Soc, 1995, 32: 66–72
14 Steck D A, Oskay W H, Raizen M G. Observation of chaos-assisted tunneling between islands of stability. Science, 2001, 5528(293): 274–278
15 Artur A, Mikhail L. Hausdorff dimension and conformal measures of feigenbaum julia sets. J Amer Math Soc, 2008, 21: 305–363
16 Zhang Z F, Ding T R, Huang W Z, et al. The Qualitative Theory of Ordinary Differential Equation (in Chinese). Beijing: Science Press, 1985
17 Camlibel K M, Pang J S, Shen J. Lyapunov stability of complementarity and extended systems. SIAM J Optim, 2006, 17: 1056–1101
18 Carbone A. Algorithm to estimate the hurst exponent of high-dimensional fractals. Phys Rev E, 2007, 76(5): 056703
19 Bandt C, Hung N V, Rao H. On the open set condition for self-similar fractals. Proc Amer Math Soc, 2006, 134(5): 1369–1374
20 Sukumar N, Wets R J -B. Deriving the continuity of maximum-entropy basis functions via variational analysis. SIAM J Optim, 2007, 18(3): 914–925
21 Krawitz P, Shmulevich I. Entropy of complex relevant components of boolean networks. Phys Rev E, 2007, 76(3): 036115
22 George E O. A property of the logistic distribution. SIAM Review, 1995, 37(4): 608–609
23 Zhang Z F, Li C Z, Zheng Z M, et al. Bifurcation Theory in Vector Fields (in Chinese). Beijing: Higher Education Press, 1997. 242–271
24 Ding J, Zhou H H. Invariant measures and their computation (in Chinese). Adv Math, 1998, 27(4): 309–323
25 Lasota A, Yorke J. On the existence of invariant measures for piecewise monotonic transformations. Trans Amer Math Soc, 1973, 186: 481–488