An Approach of Trustworthiness Evaluation of Software Behavior Based on Multidimensional Fuzzy Attributes

Zhen Li and Junfeng Tian
College of Mathematics and Computer, Hebei University, Baoding, 071002, China
Email: lizhen_hbu@yahoo.com.cn, jftian@hbu.edu.cn

Abstract—In order to increase the accuracy of trustworthiness evaluation of software, the paper improves the traditional construction process of the expected behavior trace of software and proposes an approach of trustworthiness evaluation of software behavior based on multidimensional fuzzy attributes. First, training samples of the same monitoring point are clustered based on multidimensional fuzzy attributes to construct a more accurate expected behavior trace of software. Second, an improved weight distribution method of multidimensional fuzzy attributes is presented based on correlation coefficient and standard deviation integrated approach (CCSD) for weight distribution of attributes in multiple attribute decision making. The improved weight distribution method is suitable for one-class samples from monitoring point and it considers both the dispersion of fuzzy attribute’s value and the influence among these fuzzy attributes. Finally, experiments and analyses show that: ① the expected behavior trace of software constructed by training after clustering is more accurate than without clustering; and ② our improved weight distribution method of multidimensional fuzzy attributes has better effect of trustworthiness evaluation than CCSD and other methods of weight distribution for one-class samples.

Index Terms—software behavior, trustworthiness evaluation, fuzzy attribute, clustering, weight distribution

I. INTRODUCTION

With continuous deepening of the application of software in the sensitive fields such as finance, military affairs and economy, the requirement of software trustworthiness becomes more urgent. How to ensure high confidence of software during software development and running has become an important research direction of software theory and technology [1]. If the software behavior is always accordant with the expected behavior, we call the software is trustworthy [2]. For trustworthy software, the behavior and results can be expected and the behavior states can be monitored when it runs.

The expected behavior trace of software is usually composed of a sequence of monitoring points and events or actions causing monitoring points’ transition. The common monitoring points are system call, function module, component, etc. The fine-grained monitoring points result in a high degree of trustworthiness, but with low software running efficiency. Therefore, the monitoring points should be set by comprehensively considering the needs of the degree of trustworthiness and software running efficiency in actual application. For each monitoring point, there are a group of attributes describing the expected running situation when the software runs to it. It is very important to study the multidimensional attributes of monitoring points for the trustworthiness evaluation of software behavior.

II. RELATED WORK

Because system call is the interface provided by the operating system to access system resources, system call has become the important monitoring point of software behavior. A lot of software behavior automaton models based on system call have appeared, such as finite state automaton (FSA) model and pushdown automaton (PDA) model built by static analysis of source code [3], Vt-Path model built by dynamic learning [4], HPDA model combining static analysis with dynamic learning [5], context-sensitive Dyck model [6], HFA model with static-dynamic hybrid approach [7] and the model with dataflow analysis [8]. These models can describe the running trace of software and some of them introduce arguments policies and context attributes for system call. Jones [9] and Pu et al. [10] involve time interval attribute of system calls as the criteria for determining abnormal from normal signatures. In our previous work [11], we evaluate the trustworthiness of software by several attributes of monitoring points, such as function, arguments policies, context, timestamp, memory occupancy rate and CPU occupancy rate. Li et al. [12] monitor the software behavior from the aspects of availability, reliability and security and consider many attributes such as CPU occupancy rate, IP transmission efficiency, memory occupancy rate, bandwidth utilization, average throughput, and the number of illegal connections.

The above attributes of software monitoring point can be divided into two categories: deterministic attributes...
and fuzzy attributes. For deterministic attributes, once any of them deviates from the normal value, the monitoring point is determined to be untrustworthy directly. These attributes include function, arguments, policies, context, etc. Fuzzy attributes cannot be expressed as accurate numbers. They are fuzzy and granted the prescribed error bounds. These attributes include CPU occupancy rate, IP transmission efficiency, memory occupancy rate, time interval, etc. The above references construct the expected behavior of software monitoring points by running the software many times in the training phase, and take all samples of the same monitoring point as one training set. The sample values of a fuzzy attribute for the same monitoring point can vary obviously because the path from last monitoring point to the current monitoring point can vary. Therefore, the expected behavior trace trained by the above approaches is inaccurate.

According to the feature of deterministic attributes, weight distribution is meaningless for deterministic attributes. If all the deterministic attributes of the monitoring point are trustworthy, the trustworthiness of monitoring point depends on its fuzzy attributes and the weight distribution method of fuzzy attributes determines the accuracy of trustworthiness evaluation directly for monitoring point and then for software. The common methods can be grouped roughly into three categories: subjective [13-15], objective [16,17] and integrated [18,19]. Because it is difficult to determine the weights of fuzzy attributes for monitoring point from the subjective experience, we mainly consider the objective methods. Li et al. [12] determine the weight of attributes by information entropy theory which makes the results be objective and adaptive relatively. Wang et al. [20] propose a feature weight learning algorithm which gives each feature a weight by minimizing the feature evaluation index through gradient descent technique. Wang et al. [21] propose a correlation coefficient and standard deviation integrated approach (CCSD) for determining the weights of attributes in multiple attribute decision making. The training samples for weight distribution used in these approaches either belong to different classes or are alternative schemes for multiple attribute decision making. While, the training samples for one monitoring point belong to the same class, the above methods cannot be used to the weight distribution of one-class samples. The methods of weight distribution for one-class samples are less currently. Information entropy [22,23] can be a solution and its basic idea is to determine the weight of each attribute according to the attribute’s information entropy for one-class samples. The smaller information entropy means that the sample dispersion of attribute value and the influence among these attributes are considered, which ensures the better effect of trustworthiness evaluation.

III. CLUSTERING BASED ON MULTIDIMENSIONAL FUZZY ATTRIBUTES

In the training phase, the sample values of a fuzzy attribute for the same monitoring point can vary obviously because the path from last monitoring point to the current monitoring point can vary. Therefore, the expected behavior trace constructed is inaccurate if all samples of the same monitoring point are taken as one training set. We solve the problem by clustering these samples based on multidimensional fuzzy attributes.

For n samples \(X_1, X_2, ..., X_n\) of monitoring point \(mp\), each sample has m fuzzy attributes \(A_i, A_2, ..., A_m\), denoted by sample matrix \(X = (x_{ij})_{n \times m}\) (1 ≤ \(i ≤ n, 1 ≤ j ≤ m\)). These samples are clustered by a clustering algorithm based on entropy [24].

For monitoring point \(mp\), the clustering process of \(n\) samples is as follows:

1. Number each sample as a class \(C_i\), where 1 ≤ \(k ≤ m\) and \(C_i\) is composed of \(N_i\) samples.
2. Choose any sample, add it to another class, and compute the entropy of \(C_i\) after adding the sample to it according to (1) (\(h\) is the smoothing coefficient). The sample is allocated to the class which has the smallest added value of entropy.
3. Number the classes again after allocating samples.
4. Repeat 2 and 3 until there is no class which has only one sample.
5. Compute the between-class entropy of any two classes according to (2) and merge two classes with smallest between-class entropy. The number of classes and samples for each class must be also modified.

\[H(C_i, C_j) = -\log \left( \frac{1}{(2\pi)^{\frac{m}{2}} N_i^{\frac{1}{2}} h^{\frac{m}{2}}} \sum_{i \in C_i} \sum_{j \in C_j} \exp \left( -\frac{(x_i - x_j)^2}{2h^2} \right) \right) \]  (1)

\[M(x_i, x_j)\] in (2) is as follows:

\[M(x_i, x_j) = \begin{cases} 1, & x_i \in C_i, x_j \in C_j \text{ or } x_i \in C_i, x_j \in C_j \\ 0, & \text{else} \end{cases} \]  (3)

Repeat 5 until there is only one class. If the changes of the smallest between-class entropy at a certain time are
more significant than before, the classes at this time, represented as $C_1, C_2, \ldots, C_M$ ($M$ is the number of classes), are the result of clustering.

IV. TRUSTWORTHINESS EVALUATION OF SOFTWARE BEHAVIOR

A. Trustworthy Degree of Training Sample

For the training samples $Y_1^q, Y_2^q, \ldots, Y_n^q$ of class $C_q (1 \leq q \leq M)$, let $y_{ij}^q, y_{ij}^2, \ldots, y_{ij}^m$ be $n'$ sample values of the fuzzy attribute $A_j$. After removing various effects of the environment, fuzzy attributes are approximately normally distributed [25] and the normal value of each fuzzy attribute fluctuates around the average value. Therefore, we can determine the trustworthy degree of fuzzy attribute according to the degree of deviation from the average value. The average value $\mu_j^{q}$ of fuzzy attribute $A_j$ is as follows:

$$\mu_j^{q} = \frac{\sum_{i=1}^{n'} y_{ij}^q}{n'}.$$  \hspace{1cm} (4)

The trustworthy degree $d_j^q$ of fuzzy attribute $A_j$ for training sample $Y_i^q$ ($1 \leq i \leq n'$) is as follows:

$$d_j^q = \begin{cases} \frac{y_{ij}^q}{\mu_j^{q}}, & y_{ij}^q \in [y_{\min}^q, \mu_j^{q}] \\ 1 + \frac{\mu_j^{q}}{y_{\max}^q} - \frac{y_{ij}^q}{y_{\max}^q}, & y_{ij}^q \in [\mu_j^{q}, y_{\max}^q] \end{cases},$$  \hspace{1cm} (5)

where $y_{\min}^q = \min_{l \in 1:n'} \{ y_{ij}^q \}$, $y_{\max}^q = \max_{l \in 1:n'} \{ y_{ij}^q \}$, $j = 1, 2, \ldots, m$.

From (5), we can see $d_j^q \in [0, 1]$. The trustworthy degree $d_j^q$ of training sample $Y_i^q$ is $d_j^q = \sum_{i=1}^{n'} w_i^q d_{ij}^q$, where $w_i^q$ is the weight of fuzzy attribute $A_j$ for class $C_q$.

B. Trustworthiness Evaluation

In the test phase, for the test sample $Z$ of software monitoring point $mp$, let $z_1, z_2, \ldots, z_m$ be the values of $m$ fuzzy attributes. The trustworthy degree $d_j^q$ of fuzzy attribute $A_j$ for class $C_q$ is as follows:

$$d_j^q = \begin{cases} \frac{z_j}{\mu_j^q}, & z_j \in [y_{\min}^q, \mu_j^q] \\ 1 + \frac{\mu_j^q}{y_{\max}^q} - \frac{z_j}{y_{\max}^q}, & z_j \in [\mu_j^q, y_{\max}^q] \end{cases} \quad \text{if } 0, \text{ else}$$  \hspace{1cm} (6)

The trustworthy degree $d_\alpha^q$ of test sample $Z$ for class $C_q$ is: $d_\alpha^q = \sum_{j=1}^{M} \sum_{i=1}^{n'} w_i^q d_{ij}^q$. For software monitoring point $mp$, the trustworthy degree $d$ of test sample $Z$ is: $d = \max_{l \in \{1, \ldots, C\}} (d_\alpha^q)$.

For given threshold of trustworthy degree $\tau$, if $d \in [\tau, 1]$, the software monitoring point $mp$ is trustworthy; if $d < \tau$, $mp$ is untrustworthy. The degree of untrustworthiness is increased as the deviation degree from 1 increases. When any software monitoring point is reported to be untrustworthy, the software stops running.

C. An Improved Weight Distribution Method of Multidimensional Fuzzy Attributes (Improved CCSD)

For the clustered samples, the weights of fuzzy attributes are very important to the accuracy of trustworthiness evaluation for monitoring point. CCSD method [21] for determining the weights of attributes in multiple attribute decision making considers attributes with big standard deviations should be given more important weights than those attributes with small standard deviations. However, this property is not suitable for the weight distribution of fuzzy attributes of monitoring points. We solve the problem and give an improved CCSD method for monitoring points.

For the training samples $Y_1^q, Y_2^q, \ldots, Y_n^q$ of class $C_q$, the standard deviation $\sigma_j^q$ of fuzzy attribute $A_j$ is as follows:

$$\sigma_j^q = \frac{\sqrt{\sum_{i=1}^{n'} (y_{ij}^q - \mu_j^q)^2}}{(n'-1)}. \hspace{1cm} (7)$$

The smaller standard deviation means that the sample data is more centralized, and the ability to describe normal behavior is stronger, so the weight of fuzzy attribute $A_j$ is larger, and vice versa.

When fuzzy attribute $A_j$ is dropped out, the trustworthy degree $d_\alpha^q$ of sample $Y_i^q$ is:

$$d_\alpha^q = \sum_{k=1, k \neq j}^{M} w_i^q d_{ik}^q. \hspace{1cm} \text{The correlation coefficient } R_j^q \text{ between } d_j^q \text{ and } d_\alpha^q \text{ is as follows:}$$

$$R_j^q = \frac{\sum_{i=1}^{n} (d_j^i - \bar{d}_j)(d_\alpha^i - \bar{d}_\alpha)}{\sqrt{\sum_{i=1}^{n} (d_j^i - \bar{d}_j)^2 \cdot \sum_{i=1}^{n} (d_\alpha^i - \bar{d}_\alpha)^2}}, \hspace{1cm} (8)$$

where $\bar{d}_j = \frac{1}{n'} \sum_{i=1}^{n'} d_{ij}^q$, $\bar{d}_\alpha = \frac{1}{n'} \sum_{i=1}^{n'} d_{\alpha i}^q$.

If $R_j^q$ is high enough and close to one, then fuzzy attribute $A_j$ has little effect on the trustworthiness of the sample and it can therefore be assigned a very small weight. If $R_j^q$ is very low, say close to minus one, then fuzzy attribute $A_j$ has significant impact on the
trustworthiness of the sample and it should be given a
very important weight.

We integrate the standard deviation and correlation
coefficient and define the weights of fuzzy attributes as follows:

\[
w_j^* = \frac{\sqrt{1 - R^*_j}}{\sigma^*_j} - \frac{\sum_{k=1}^{m} \sqrt{1 - R^*_k}}{\sigma^*_k}.
\]  

Equation (9) is a system of nonlinear equations. It
contains \( m \) equations, which can uniquely determine \( m \)
weight variables. To solve the equation, we convert it into
the following nonlinear optimization model for solution:

\[
\text{Minimize } J = \sum_{j=1}^{m} \left( w_j^* \cdot \text{error} \right)^2,
\]  

subject to \( \sum_{j=1}^{m} w_j^* = 1, \ w_j^* \geq 0 \), which can be solved by LINGO
software and at optimality the objective function value
\( J^* = 0 \).

V. EXPERIMENTS AND ANALYSIS

A. Test of Clustering Based on Multidimensional Fuzzy
Attributes

We have performed experiments on a PC with Intel (R)
Core (TM)2 Duo E7500 2.93 GHz and 2 GB of main
memory running Linux kernel 2.4.20. The grain of
monitoring point is set to system call. Each system call is
intercepted by loadable kernel module (LKM) and
modified to capture the attributes' values of the system
call. Because the trustworthiness of deterministic attributes
determines the trustworthiness of monitoring point
directly, we mainly discuss the effect of fuzzy attributes
on the trustworthiness of monitoring point.

Software vi6.1 is the editor in Red Hat 9 Linux. In clean
environment, we capture 15 samples along three different
paths in a monitoring point of software vi6.1 where three
paths converge. Three fuzzy attributes time interval, memory
variation and CPU variation are involved and they are the
absolute value of difference of time, memory occupancy rate
and CPU occupancy rate respectively between the current
monitoring point and last monitoring point, as shown in
Table I. Table II shows the fuzzy attributes' values of two
normal traces and two abnormal traces in the monitoring
point for test.

The experiment clusters the training samples based on
multidimensional fuzzy attributes. Let smoothing
coefficient \( h \) be 0.5. The result of clustering is:
Class 1: \{1, 2, 3, 4, 5, 6\}, Class 2: \{7, 8, 9, 10\}, Class 3: \{11, 12, 13, 14, 15\}. Suppose the weight of each fuzzy attribute is equal and
let the threshold of trustworthy degree \( \tau \) be 0.85, the
trustworthy degrees of test samples without clustering and
after clustering for each class are shown in Table III. After clustering, the trustworthy degree of test sample is
the maximum of the trustworthy degrees for all classes,
that is, the trustworthy degree of Normal 1, Normal 2,
Abnormal 1 and Abnormal 2 is 0.95, 0.97, 0.70 and 0.67
respectively. Without clustering, two normal traces are
determined to be untrustworthy by mistake. While, after
clustering, the trustworthiness of both two normal traces
and two abnormal traces are determined correctly.
Therefore, the expected behavior trace of software is
constructed more accurately by training after clustering.

B. Comparison of Weight Distribution Methods

Our experiment uses machine learning databases (MLDBs)
from UCI Repository [26] including lots of different
databases. We select Iris database which has three classes
(setosa, versicolor and virginica). Each class has 50 samples
each sample has four attributes (sepal length, sepal width,
petal length and petal width) which are all continuous real
variables. There is a great difference between the attributes'
values of class setosa and other two classes; While, there is
little difference between the attributes’ values of class
versicolor and class virginica.

Let’s take class setosa and class versicolor for example.
For each class, we take the first forty-five samples as
training samples and the last five samples as test samples.
Fig. 1 and Fig. 2 show the weights of each fuzzy
attributes computed with four weight distribution
methods (equal weight, CCSD [21], information entropy
[23] and improved CCSD) for class setosa and class
two classes, all test samples of class setosa are determined to be trustworthy and all samples of class versicolor and virginica are determined to be untrustworthy correctly for the four weight distribution methods. In the second experiment, we take the training samples of class versicolor as training samples and take the test samples of class versicolor, all samples of class setosa and virginica as test samples. The test results of four weight distribution methods are shown in Table IV. For the test samples of class versicolor, the test results of four weight distribution methods are same. For samples of class setosa, the evaluation results of four weight distribution methods are same and have no misjudgment samples; For the average of trustworthy degree of four weight distribution methods, improved CCSD has the smallest average of trustworthy degree which means the largest degree of deviation, so it can determine the untrustworthy sample most easily. For samples of class virginica, the evaluation results of improved CCSD have the smallest misjudgment samples and the smallest average of trustworthy degree; the second is information entropy. CCSD for determining the weights of attributes in multiple attribute decision making is the worst one. Therefore, our improved CCSD has better effects for trustworthiness evaluation.

VI. CONCLUSION

The paper proposes an approach of trustworthiness evaluation of software behavior based on multidimensional fuzzy attributes. It clusters the samples of the same monitoring point based on multidimensional fuzzy attributes to construct a more accurate expected trace of software. For a better trustworthiness evaluation effect, the paper proposes an improved weight distribution method of multidimensional fuzzy attributes based on CCSD method for weight distribution of attributes in multiple attribute decision making. The improved weight distribution method is suitable not only for the multidimensional fuzzy attributes of monitoring points, but also for any weight distribution according to one-class samples, so it is of widespread usage. Our future work is to consider the selection of multidimensional fuzzy attributes for monitoring points to achieve a better effect for trustworthiness evaluation of software.

REFERENCES


Zhen Li, born in Baoding, China, 1981, received M.S. degree of computer application from Hebei University, Baoding, China in 2006. She is a lecturer of the College of Mathematics and Computer of Hebei University. In the past five years, she has published over 10 technical papers in refereed journals and conference proceedings. Her current research interests include computer security, trust computing and distributed computing.

Junfeng Tian, born in Baoding, China, 1965, received Ph.D degree of computer science from University of Science and Technology of China, Hefei, China in 2004. He is a professor of Computer Science at Hebei University. In the past five years, he has published over 30 technical papers in refereed journals and conference proceedings. His research interests include network security, trust computing, distributed computing and wireless sensor network.

More information about Prof. Tian can be found at http://int.hbu.edu.cn/int/szll/tjf.html.