基于检查点场景信息的软件行为可信预测模型

田俊峰[1,2]，郭玉慧[1,2]
（1. 河北大学网络空间安全与计算机学院，河北 保定 071002；2. 河北省高可信信息系统重点实验室，河北 保定 071002）

摘 要：为了保证软件可信性，通过动态监测软件行为，对软件在一段时间内运行的可信状态进行评估，提出了一种基于检查点场景信息的软件行为可信预测模型 CBSI-TM。该模型通过在软件运行轨迹中设置若干检查点，并引入相邻检查点时间增量和 CPU 利用率变化量定义场景信息，用以反映相邻检查点场景信息的关系，然后利用径向基函数（RBF, radial basis function）神经网络分类器评估当前检查点的状态来判断软件的可信情况，并运用半加权马尔可夫模型预测下一个检查点的状态，达到对软件未来运行趋势的可信情况的评估。实验结果证明了 CBSI-TM 模型能够有效地预测软件未来运行趋势的可信情况，并验证了该模型具有更优的合理性和有效性。

关键词：软件可信性；检查点；RBF 神经网络；半加权马尔可夫模型
中图分类号：TP393.08
文献标识码：A
doi: 10.11959/j.issn.1000−436x.2018163

Software behavior trust forecast model based on check point scene information

TIAN Junfeng[1,2], GUO Yuhui[1,2]
1. School of Cyber Security and Computer, Hebei University, Baoding 071002, China
2. Key Lab on High Trusted Information System in Hebei Province, Baoding 071002, China

Abstract: In order to ensure the trustworthiness of software, and evaluate the trusted status of the software after running for a period of time by monitoring software behavior dynamically, a software behavior trust forecast model on checkpoint scene information which was called CBSI-TM was presented. The model set up a number of checkpoints in the software running track, and introduced the time increment of adjacent checkpoints, and the change of CPU utilization rate to define the scene information, and reflected the relationship between adjacent checkpoints scene information. Then the RBF neural network classifier evaluated the status of the current checkpoint to judge the trustworthiness of the software, and the semi weighted Markov model predicted the situation of the next checkpoint to evaluate the trustworthiness of future running trend of the software. The experimental results show that the CBSI-TM model can predict the future trusted status of the software effectively, and verify that the model is more reasonable and effective.

Key words: software trustworthiness, check point, RBF neural network, semi weighted Markov model

1 引言

软件可信一直是人们普遍关注的问题。软件可信是指软件系统能够按照其设定目标所预期的方式运行，软件行为和用户的预期相一致[1]。由于软件本身的设计缺陷或软件运行环境的变化可能导致软件运行故障，从而偏离预期行为轨迹，最终给人们的工作和生活带来不良影响甚至造成巨大损失。因此，研究软件行为的可信性具有重要意义。

当前对软件行为的可信研究已有不少成果。李

收稿日期：2018−01−09；**修回日期**：2018−07−30
基金项目：国家自然科学基金资助项目（No.61170254）；河北省自然科学基金资助项目（No.F2016201244）
Foundation Items: The National Natural Science Foundation of China (No.61170254), The Natural Science Foundation of Hebei Province (No.F2016201244)

珍等[2]采取伴随式分布软件监控机制，在节点内置入检查点，引入适合复杂交互场景的交互关联方法，提出了检查点可信性及检查点之间结构可信性的评估方法。刘玉玲等[3]提出一种基于软件行为的检查点风险评估信任模型，通过累积多个有疑似风险的检查点，运用风险评估策略，判定有疑似不可信的检查点，利用处罚或奖励机制求出软件行为的可信度，最终判别软件行为是否可信。吴佳等[4]提出了一种基于隐马尔可夫模型的软件系统状态预测方法，该方法通过收集系统的外在特征信息，利用隐马尔可夫模型建立系统内部状态与外部特征之间的联系，实时了解并预测系统状态。Guo 等[5]提出了一个新的统一框架来表示硬件基础设施和软件程序，在统一框架的支持下，系统设计者/集成商将能够从不可信的第三方供应商正式验证与硬件和软件集成的计算机系统的可信性。王德鑫等[6]根据从软件开发过程的实体、行为以及制品 3 个方面获取的可信证据，提出了由 37 个可信规则、182 个过程可信证据和 108 个制品可信证据组成的软件过程可信度模型，并给出了基于该模型证据的软件过程可信评价方法。贾晓辉等[7]提出了软件质量模型及分级的可信软件评估模型，将软件的信任程度分为 6 个可信级别，基于决策树给出了可信软件等级的评价过程，并将其运用在可信构件平台中。王犇等[8]提出了一种基于多属性熵权合成的软件可信等级评估方法，该方法重点解决了软件可信性评估中可信证据合成时证据冲突、多属性权重分配等问题，从而使软件可信评估的结果更加准确和真实。Li 等[9]提出了一个可靠性评估模型，着重分析了软件可靠性的不同组成部分的影响，利用复杂网络理论中最有影响力的节点发现算法来计算每个组件的影响因素，根据影响因素对软件系统的可靠性进行了评估。Okamura 等[10]提出了一个开放源码软件可靠性评估的统一建模框架，结合经典的非齐次泊松过程的软件可靠性增长模型（SRGM, software reliability growth model）与一个熟悉的回归方案——广义线性模型（GLM, generalized linear model），发展了一种新的框架，不仅可以估计软件的可靠性，还可以研究软件度量对故障检测过程的影响。Liu 等[11]构建了基于神经网络的软件可靠性的评价指标体系，并最终实现了软件可靠性评估系统。这些研究大多是通过分析软件行为来评估当前软件系统的可信性，而通过针对软件当前运行节点可信情况来预测下一个节点的行为可信性，即对软件未来的运行趋势进行评估的研究相对较少。

因此，本文提出了一种基于检查点场景信息的软件行为可信预测模型，该模型的主要贡献如下。

1) 将获取的检查点场景信息输入训练好的 RBF 神经网络，对当前检查点处的软件行为进行可信性评估。

2) 在检查点场景信息中，引入了时间增量和 CPU 利用率变化量，反映相邻检查点之间的系统状态变化情况。

3) 结合当前检查点的场景信息和 RBF 对其可信性的评估结果，进一步运用改进的半加权马尔可夫模型预测下一个检查点的可信情况。

4) 实验结果表明，该模型可以针对性地预测软件未来运行趋势的可信性，便于及时管理软件行为，加强软件的可信性。

2 软件行为可信预测模型

2.1 相关定义

定义 1 检查点（check point）。软件流程上的一些关键点，一般选取软件独立功能结束处和软件分支处或比较重要的系统调用处作为检查点，用以提取软件的场景信息。

定义 2 场景（scene）。检查点处必要的软件运行背景信息和结果信息，包括系统调用、系统调用参数、计算结果（转移条件）、CPU 利用率、内存占用率、软件运行时间等，它是一个 n 元组，记为 $\boldsymbol{Y}$，$\boldsymbol{Y}=\{x_1,x_2,\cdots,x_n\}$。

定义 3 行为轨迹。由软件的运行轨迹和功能轨迹组成，是指软件主体所实施的行为按照时间顺序记录下来形成的形式化序列。

定义 4 运行轨迹。是指按执行顺序将检查点串联起来的序列。

定义 5 功能轨迹。与运行轨迹相对应的检查点场景信息序列。

定义 6 检查点时间。从程序开始运行或从检查点开始运行到该检查点的时间，记为 $T_i\ (i=0,1,2,\cdots,n-1)$。

定义 7 检查点时间增量。第 i 个检查点与第 i–1 个检查点的时间之差，记为 $\Delta T_i=T_i-T_{i-1}\ (1\leqslant I\leqslant n-1)$。

定义 8 CPU 利用率变化量。第 i 个检查点与第 i–1 个检查点的 CPU 利用率之差，记为 $\Delta P_i=P_i-P_{i-1}\ (1\leqslant i\leqslant n-1)$。

定义 9 软件行为监测。是指软件运行时，在

检查点处对行为信息进行收集、评估等操作，一般通过监测（软件）进行。

2.2　场景信息的选择

检查点的可信评估依赖于检查点的行为信息。根据在软件运行时监测到的检查点场景信息来分析检查点行为的可信情况，当检查点行为发生异常时，场景中的很多属性信息会发生变化。吴佳等[4]选择了变化比较直观的 CPU 利用率、内存占用率和磁盘利用率等作为 WebLogic 服务器的系统特征。根据文献[4]的研究，CBSI-TM 模型选择了检查点时间 T、CPU 利用率 P、内存利用率 M、网络传输速度 B，同时为了研究相邻检查点之间发生的系统状态变化，即通过软件当前检查点的状态进而预测下一个检查点的状态，CBSI-TM 模型还选择了相邻检查点时间增量 ΔT、CPU 利用率变化量 ΔP，最终将以上 6 个影响因素作为检查点的场景信息。

检查点时间变化量和 CPU 利用率变化量反映的是第 i−1 检查点与第 i 个检查点之间的系统状态变化，通过大量的实验测试可知，程序正常运行在相同的环境下，这 2 个变化量的数据均满足正态分布 $X \sim \mathrm{N}(\mu,\sigma)$ (X 表示 ΔT 或 ΔP)，并且处于 $[\mu - n\sigma, \mu + n\sigma]$ 范围内。如果超出这个范围，可能是由于以下 3 个原因造成的：1)程序本身发生改变，出现异常；2)运行背景有其他程序与该程序并行运行；3)运行该程序的机器（包括虚拟机）出现异常。

因此，选择以上这 6 个影响因素既能够反映当前检查点的系统状态，又能够反映相邻检查点的系统状态变化。

2.3　相关方法的介绍

2.3.1　RBF 神经网络

RBF 神经网络[12]是一个具有单隐含层的 3 层前馈网络，具有非线性逼近能力强、网络结构简单、学习速度快、不易陷入局部极小点和顽健性能好等优点，利用 RBF 神经网络有助于对大量、复杂的信息进行科学的分类。RBF 神经网络结构如图 1 所示，包括输入层、隐含层和输出层。输入层节点只传递输入信息到隐含层，隐含层节点对输入信号在局部产生响应，输出层节点通常是简单的线性函数。

2.3.2　马尔可夫模型

马尔可夫过程[13]是研究某一事件的状态及状态之间转移规律的随机过程理论，它通过分析某一时刻不同状态的初始概率及状态之间转移概率的关系来研究未来某时刻状态的变化趋势。

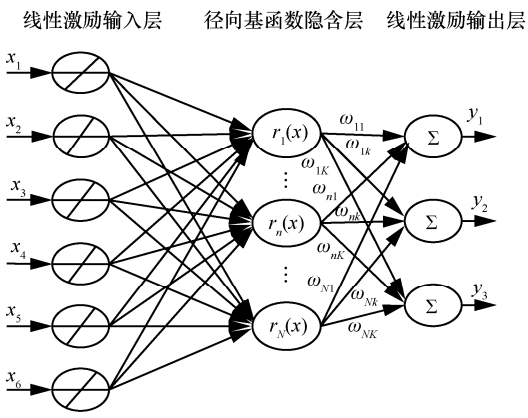


图 1　RBF 神经网络

马尔可夫链预测的理论基础是马尔可夫过程，对其运动变化的分析主要是研究链内有限马尔可夫过程的状态及其相互关系，进而预测链的未来状况，并据此做出决策。马尔可夫模型可表示为

$$\boldsymbol{x}(n) = \boldsymbol{x}(0)\boldsymbol{P}^n$$

其中，$\boldsymbol{x}(n)$为 n 时刻的状态概率向量，$\boldsymbol{x}(0)$为初始时刻的状态概率向量，$\boldsymbol{P}$ 为状态转移概率矩阵。

由 2.2 节的分析可知，影响检查点状态变化的场景信息包括检查点时间、CPU 利用率、相邻检查点时间增量、CPU 利用率变化量、内存利用率、网络传输速度，这些影响因素具有很多不确定性和随机时变特性，因此导致检查点状态发生改变时也呈现了很强的随机时变特性，并且其变化趋势只与前一个检查点状态的场景信息有关，而与其他检查点的状态无关。因此可以利用马尔可夫模型的特性来预测检查点的状态，进而对软件的异常行为及时进行调整。

2.4　软件行为可信预测框架

基于检查点场景信息的软件行为可信预测模型的框架如图 2 所示。在每个检查点都部署一个监测器，并为每个被监测检查点创建一个独立的信息存储队列。监测器获取到检查点的场景信息后，将监测数据暂存在对应的信息队列中，同时添加在本地监测信息库中。对每个检查点，设置一个 RBF 神经网络分类器，对检查点场景信息进行离线训练和在线评估，步骤如下。

步骤 1　离线训练。从本地监测信息库获取场景信息，训练 RBF 神经网络。检查点场景信息通过输入层传递到隐含层，隐含层神经元使用混合高斯激励函数对输入层的信息进行运算处理，输出层神经元对隐含层神经元的输出进行加权求和并输出

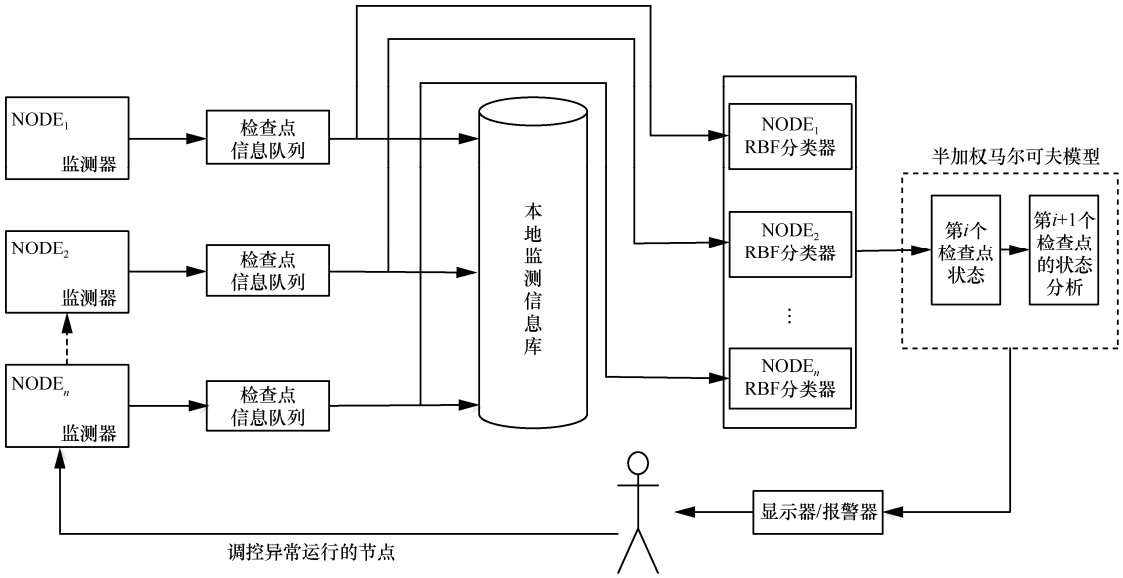


图 2 基于检查点场景信息的软件行为可信预测模型的框架

结果，训练完成后，保存训练好的 RBF 神经网络。

步骤 2 在线评估。监测器捕获软件运行时检查点的场景信息并暂存在对应信息队列，RBF 神经网络分类器从检查点信息队列获取场景信息进行处理，评估当前检查点的可信情况。

然后通过半加权马尔可夫模型，得到软件在下一个检查点处于不同状态的概率或在当前检查点进行调整的概率，分为离线训练和在线评估 2 个步骤，介绍如下。

步骤 1 离线训练。将软件运行时的第 i 个检查点与第 i+1 个检查点的状态信息作为样本集，训练得到第 i 个检查点的状态概率向量 $\boldsymbol{y}_i$，第 i 个检查点正常或临界状态 a_k 转移到第 i+1 个检查点状态 a_j 的概率 P_{kj}，用第 i 个检查点正常或临界状态对第 i+1 个检查点影响程度的权重对概率 P_{kj} 加权；如果第 i 个检查点异常，则进行人工干预，调整为正常状态的概率为 P_{kj}。对所有的转移概率进行规范化，得到半加权马尔可夫模型的转移概率矩阵，建立半加权马尔可夫模型。

步骤 2 在线评估。在软件运行时，RBF 分类器评估第 i 个检查点的状态，若异常，则在第 i 个检查点处进行人工干预检查；否则，根据第 i 个检查点的状态概率向量 $\boldsymbol{y}_i$ 与加权转移概率相乘得到第 i+1 个检查点的状态概率向量，该向量中的最大概率对应第 i+1 个检查点的可能状态。例如，检查点以较高的概率进入异常状态，通过显示器或报警器通知管理员干预检查，甚至暂停运行，进而使软件的异常运行及时得到调控。

3 基于 RBF 神经网络的检查点状态分类

人们一般采用语言量对信任等级进行描述，借鉴文献[14]的思想，该文将检查点状态划分为 3 个等级：正常状态、临界状态、异常状态。首先需要检查点场景信息样本集对 RBF 神经网络进行离线训练，然后利用训练好的 RBF 神经网络对当前检查点的状态进行评估。

步骤 1 离线训练。经过 n 个软件运行周期，利用监测器捕获检查点的场景信息，暂存在检查点队列，同时添加在本地监测信息库，组成训练样本集的输入矩阵 $\boldsymbol{X}=[x_1,x_2,\cdots,x_s,\cdots,x_S], X\in R^{M\times S}$，训练样本集的期望输出矩阵 $\boldsymbol{Y}=[y_1,y_2,\cdots,y_s,\cdots,y_S], Y\in R^{K\times S}$，然后把场景信息的样本集分为 3 个子集：训练样本集、校正样本集和测试样本集。其中，训练数据集用于训练神经网络；校正数据集用于检验在权值确定过程中神经网络的输出与期望输出的误差，即校正误差；测试数据集用于测试训练好的神经网络分类的性能。

监测器捕获的检查点场景信息有 6 类，分别是检查点时间、CPU 利用率、相邻检查点时间增量、CPU 利用率变化量、内存利用率、网络传输速度，检查点的可能状态有 3 种：正常状态、临界状态、异常状态，所以 RBF 神经网络输入层有 6 个神经元，则输入样本向量为 $\boldsymbol{x}=[x_1,x_2,x_3,x_4,x_5,x_6]^{\mathrm{T}}$，输出层神经元数目为 3，则输出向量为 $\boldsymbol{y}=[y_1,y_2,y_3]^{\mathrm{T}}$，其

中，上标 T 表示向量的转置。设置隐含层神经元数与输入的样本数相同，并将每个样本 x_i 作为一个隐含层神经元的中心，输入层与隐含层神经元之间的连接权值均设为 1，其径向基函数使用混合高斯函数，混合高斯函数[15]是聚类算法和概率密度拟合中的常用函数，其基本假设是待研究数据服从混合高斯分布，且多数自然数据具有分类聚集性质，由中心极限定理可知，检查点的场景信息服从混合高斯分布，根据高斯分布的可加性，有 $\sum_{i=1}^{M} k_i \boldsymbol{X}_i \sim \mathrm{N}_n\left(\sum_{i=1}^{M} k_i \boldsymbol{\mu}_i, \sum_{i=1}^{M} k_i^2 \boldsymbol{\Sigma}_i\right)$。检查点 e 的场景信息可表示为 $S(e) \sim \mathrm{N}_n(\mu(e), \Sigma(e))$，采用 EM 算法[16]求解混合高斯函数的参数，其中，k_i 表示训练样本与第 i 个高斯模型匹配的概率，μ_i 表示第 i 个高斯模型的均值，Σ_i 表示第 i 个高斯模型的方差，反映样本的偏差程度，则第 n 个隐含层神经元的激励函数可表示为

$$r_n(x) = \exp\left(\frac{\left\|\boldsymbol{x} - \sum_{i=1}^{n} k_i \boldsymbol{\mu}_i\right\|_2^2}{2\left(\sum_{i=1}^{n} k_i^2 \boldsymbol{\Sigma}_i\right)^2}\right) \tag{1}$$

其中，$\|\bullet\|_2$ 表示向量的二范数，$\sum_{i=1}^{n} k_i \boldsymbol{\mu}_i$ 表示混合高斯径向基函数的中心，与神经网络的输入 $\boldsymbol{x}$ 具有相同的维数，$\sum_{i=1}^{n} k_i^2 \boldsymbol{\Sigma}_i$ 表示混合高斯函数的方差。

采用基于伪逆的权值直接确定法[17]，可直接得到隐含层和输出层神经元之间的最优连接权值。在 RBF 神经网络中求解连接值时，隐含层与输出层神经元的连接权值形成矩阵 $\boldsymbol{W}$，利用式(2)确定连接权值，其中，$\boldsymbol{Q}^{+}$为激励矩阵，$\boldsymbol{Y}^{\mathrm{T}}$ 为期望输出矩阵，则有

$$\boldsymbol{W} = \boldsymbol{Q}^{+}\boldsymbol{Y}^{\mathrm{T}} \tag{2}$$

网络输出层第 m(m=1,2,3)个神经元的输出为

$$y_k = \sum_{n=1}^{N} w_{nk} r_n(x) \tag{3}$$

定义网络的平均输出误差[18]E 为

$$E = \frac{\|\boldsymbol{Y} - \boldsymbol{\Gamma}\|_{\mathrm{F}}^2}{DK} \tag{4}$$

其中，$\boldsymbol{Y}$ 表示网络的期望输出矩阵，$\boldsymbol{\Gamma}$ 表示网络的实际输出矩阵，D 表示样本数，K 表示输出类别数，$\|\bullet\|_{\mathrm{F}}$ 表示矩阵的 F 范数。

通过训练数据集训练 RBF 神经网络，计算网络的平均输出误差 $E_{\min}$，然后利用校正数据集，计算网络校正的平均输出误差 E_{cur}，并增加隐含层神经元的个数，若 $E_{\mathrm{cur}}<E_{\min}$，则令 $E_{\min}=E_{\mathrm{cur}}$；随着隐含层神经元个数的增加，若 $E_{\mathrm{cur}}>E_{\min}$，则删除新增加的隐含层神经元个数，此时网络结构已达到最优隐含层神经元数。

RBF 神经网络在经过以上步骤训练后，隐含层神经元中心、方差以及隐含层和输出层之间的连接权值 $\boldsymbol{W}$ 都被确定，这样 RBF 分类系统就建立起来了。

输入测试样本向量 $\boldsymbol{x}_t$，可计算得到对应的网络输出 $\boldsymbol{y}_t$，然后对 $\boldsymbol{y}_t$ 进行分类处理：若 $y_{tk} = \max\{y_{t1}, y_{t2}, \cdots, y_{tK}\}$，则取 y_{tk}=1；否则，取 y_{tk}=0。判断分类处理后向量所属的类别时，若处理后的输出向量为 $\boldsymbol{y}_t = [10\cdots0]$，则属于第一类；若处理后的输出向量为 $\boldsymbol{y}_t = [01\cdots0]$，则属于第二类，依次类推。

步骤 2　在线评估。利用训练好的 RBF 神经网络分类器，就可以将监测器捕获到运行时的检查点场景信息输入 RBF 神经网络，评估该检查点的可信情况，若检查点状态正常或临界，则利用半加权马尔可夫模型预测下一个检查点的可信情况，否则通知管理员进行干预，甚至暂停运行。

4　基于半加权马尔可夫的检查点状态预测

4.1　检查点的场景值计算

由于场景信息中的各个因素对检查点状态的影响程度不同，因此半加权马尔可夫模型对各个属性因素采用加权的方法进行计算，求得综合场景值。采用模糊层次分析法 FAHP[19]，求得各场景信息的权重 ω，则检查点 e 场景值计算式为

$$S(e) = g_{\mathrm{T}}\omega_1 + g_{\Delta\mathrm{T}}\omega_2 + g_{\mathrm{P}}\omega_3 + g_{\Delta\mathrm{P}}\omega_4 + g_{\mathrm{M}}\omega_5 + g_{\mathrm{B}}\omega_6 \tag{5}$$

其中，$S(e)$ 表示检查点 e 的场景值，g_{T} 表示检查点时间，$g_{\Delta\mathrm{T}}$ 表示相邻检查点时间增量，g_{P} 表示 CPU 利用率，$g_{\Delta\mathrm{P}}$ 表示相邻检查点 CPU 利用率变化量，g_{M} 表示内存利用率，g_{B} 表示网络传输速度。

半加权马尔可夫模型依据各场景信息的相对重要程度不同，在计算场景信息的权重时数量标度如表 1 所示。

表 1 数量标度

标度	定义	说明
Ⅰ	同等重要	2 个场景信息因素的重要性相当
Ⅱ	稍微重要	2 个场景信息因素的重要性差不多
Ⅲ	明显重要	2 个场景信息因素的重要性差异相对显著
Ⅳ	重要得多	2 个场景信息因素的重要性差异非常明显
Ⅴ	极端重要	2 个场景信息因素的重要性处于 2 个极端
Ⅵ	反比较	若因素 a_i 与 a_j 比较得到判断 r_{ij}，则 $r_{ji}=1-r_{ij}$

根据表 1，按场景信息元素的重要程度，通过两两比较，建立优先关系矩阵[3]$\boldsymbol{A}$ 为

$$\boldsymbol{A}=\begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \cdots & r_{nn} \end{bmatrix}$$

然后将优先矩阵转化成一致模糊矩阵，对优先关系矩阵 $\boldsymbol{A}$ 按行求和，记为 $r_i=\sum_{j=1}^{m} r_{ij}$,$i=1,2,\cdots,n$，进行 $r'_{ij}=\frac{r_i-r_j}{2n}+0.5$ 变换，然后依据式(6)计算权重[3] ω_k 为

$$\omega_k=\frac{1}{n}-\frac{1}{2a}+\sum_{j=1}^{m}\frac{r'_{ij}}{na},k=1,2,\cdots,m,a\geqslant\frac{n-1}{2} \quad (6)$$

由于检查点 i 的不同状态对检查点 i+1 的影响程度不同，因此通过式(7)表示检查点 i+1 的状态，即

$$S(i+1)_k=S(i)_1\mu_i+S(i)_2\mu_j$$
$$(i,j)=\{(1,2),(3,4),(5,6)\},k=1,2,3 \quad (7)$$

其中，$S(i)_1$、$S(i)_2$ 分别表示检查点 i 的正常状态和临界状态，μ_1、μ_2 分别表示检查点 i 的正常状态和临界状态对检查点 i+1 的正常状态影响程度的权重，μ_3、μ_4 分别表示检查点 i 的正常状态和临界状态对检查点 i+1 的临界状态影响程度的权重，μ_5、μ_6 分别表示检查点 i 的正常状态和临界状态对检查点 i+1 的异常状态影响程度的权重。向量 $\boldsymbol{S}_i=\{\overline{S_1},\overline{S_2}\}$ 表示检查点 i 正常、临界的平均场景值，$\boldsymbol{S}_{i+1}=\{\overline{S_{11}},\overline{S_{12}},\overline{S_{13}},\overline{S_{21}},\overline{S_{22}},\overline{S_{23}}\}$ 表示检查点 i+1 的平均场景值，其中，$\overline{S_{1n}}$ 表示检查点 i 正常时，检查点 i+1 的正常、临界、异常的平均场景值，$\overline{S_{2n}}$ 表示检查点 i 临界时，检查点 i+1 的正常、临界、异常的平均场景值。比较 $\frac{\overline{S_1}}{\overline{S_{1n}}}$ 与 $\frac{\overline{S_2}}{\overline{S_{2n}}}$ 的大小，$\frac{\overline{S_1}}{\overline{S_{1n}}}$ 比 $\frac{\overline{S_2}}{\overline{S_{2n}}}$ 越大，表示检查点 i 的正常状态比临界状态对检查点 i+1 的状态影响程度越大，反之，表示对检查点 i+1 的状态影响程度越小。

在求解权重 μ 时，首先要将检查点 i 的正常状态和临界状态对检查点 i+1 影响程度大小进行比较，然后建立优先关系矩阵，转换成模糊一致矩阵，利用式(6)计算权值 μ。

4.2 半加权马尔可夫模型的确定

半加权马尔可夫模型设定了 4 个状态，其状态空间 E={1, 2, 3, 4}，其转移概率矩阵由 2 种概率组成：1) 当前检查点正常或临界状态转移到下一个检查点任一状态的概率加权重 μ 规范化后作为加权转移概率；2) 当前检查点异常状态进行调整的概率规范化后作为调整转移概率。将系统在第 i 个检查点处于状态 k 转移到第 i+1 个检查点处于状态 j 的状态转移概率和检查点 i 异常状态调整的概率记为 p_{kj}，$k,j\in E$。

当第 i 个检查点处于正常或临界状态时，若用 f_{kj} 表示检查点从第 i 个检查点状态 k 经过转移到达第 i+1 个检查点状态 j 的频数，那么由 f_{kj} 组成的矩阵称为转移频数矩阵，转移频数 f_{kj} 除以所在行总和即为从第 i 个检查点状态 k 经过转移到达第 i+1 个检查点状态 j 的转移概率，对转移概率加权重 μ，构成加权转移概率，其中，$p_{kj}=\frac{\mu_j f_{kj}}{\sum_{j=1}^{3} f_{kj}}$，$k=1,2$，$j=1,2,3$ 且 $p_{kj}>0$。

当第 i 个检查点处于异常时，就通知管理员对检查点 i 进行调控，检查点 i 从异常状态转移到干预状态，经过调整，检查点 i 从干预状态转为正常状态。所以检查点 i 从异常状态转移到干预状态和从干预状态转移到正常状态是必然事件，其概率 p_{34}、p_{41} 都为 1，而从异常状态转移到其他状态是不可能事件，其概率 p_{33} 为 0。

对第 i 个检查点所有状态转移概率 p_{kj}，通过式(8)进行规范化变换，有

$$p'_{kj}=\frac{p_{kj}}{\sum_{j=1}^{3} p_{kj}},\ k,j\in E \quad (8)$$

规范化变换后，所有的转移概率 p'_{kj} 构成半加权马尔可夫模型转移概率矩阵 $\boldsymbol{P}'$ 为

$$\boldsymbol{P}'=\begin{bmatrix} p'_{11} & p'_{12} & p'_{13} \\ p'_{21} & p'_{22} & p'_{23} \\ p'_{34} & p'_{41} & p'_{33} \end{bmatrix}$$

其中，p'_{1j} (j=1,2,3)表示第 i 个检查点是正常状态时，转移到第 i+1 个检查点的状态是正常状态、临界状态、异常状态的概率，p'_{2j} (j=1,2,3)表示第 i 个检查点是临界状态时，转移到第 i+1 个检查点的状态是正常状态、临界状态、异常状态的概率，p'_{34} 表示第 i 个检查点从异常状态转移到干预状态的概率，p'_{41} 表示第 i 个检查点从干预状态转移到正常状态的概率，p'_{33} 表示从异常状态转移到其他状态的概率。

通过以上过程即可建立初始半加权马尔可夫模型，如图 3 所示。

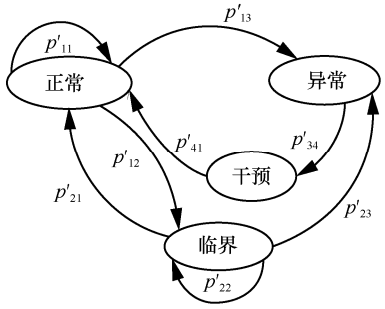


图 3　半加权马尔可夫状态转移模型

4.3　检查点的可信预测

为了使预测有意义，预测应满足如下约束条件：对第 i+1 个检查点的状态预测的时间开销为 t_{i+1}^{p}，检查点 i 和检查点 i+1 时间之差为 $\Delta t_{i+1}=t_{i+1}-t_i$，$t_{i+1}^{p}<\Delta t$。

根据收集的检查点 i 的场景信息，利用 RBF 分类器判定检查点的状态，可以确定检查点 i 的状态概率向量为 $\boldsymbol{y}_i=(y_1,y_2,y_3)$，$\boldsymbol{y}_i$ 的各元素值就是检查点 i 处于各状态的概率值，且满足 $\sum_{n=1}^{3}y_n=1$。

检查点 i+1 的状态概率向量可表示为 $\boldsymbol{y}_{i+1}=(y_1,y_2,y_3)$。因为检查点的概率向量中各元素值就是检查点处于各状态的概率值，所以当检查点 i 处于正常或临界状态时，可以通过式(9)计算检查点 i+1 的状态概率向量，其中，状态概率的最大值所对应的状态即为第 i+1 个检查点的预测状态。

$$\boldsymbol{y}_{i+1}=\boldsymbol{y}'_i\boldsymbol{p}' \tag{9}$$

其中，$\boldsymbol{y}'_i=(y_1,y_2)$，$\boldsymbol{p}'=\begin{bmatrix}p'_{11} & p'_{12} & p'_{13}\\ p'_{21} & p'_{22} & p'_{23}\end{bmatrix}$。

预测过程如下。

1) 监测器捕获软件运行时第 i 个检查点的场景信息，然后通过 RBF 神经网络分类器判断检查点 i 的状态。

2) 若检查点 i 处于异常状态，通过显示器或报警器通知管理员干预检查，甚至暂停运行。

3) 若检查点 i 处于正常状态，则计算检查点 i 的场景值 S_1，比较 $\frac{S_1}{S_{1n}}$ 与 $\frac{\overline{S_2}}{S_{2n}}$ (n=1,2,3)大小；若检查点 i 处于临界状态，则计算检查点 i 的场景值 S_2，比较 $\frac{\overline{S_1}}{S_{1n}}$ 与 $\frac{S_2}{S_{2n}}$ (n=1,2,3)大小。根据式(6)修改权值 μ，对原始未加权转移矩阵重新加权，然后规范化，构成新的半加权马尔可夫模型的转移概率矩阵。

4) 若检查点 i 处于正常或临界状态，则通过式(9)计算检查点 i+1 的状态概率向量 $\boldsymbol{y}_{i+1}$，其中，状态概率的最大值所对应的状态即为第 i+1 个检查点的预测状态；若显示为异常，则通过显示器或报警器通知管理员干预检查，甚至暂停运行，否则，不进行任何调整。

5　实验与分析

为了验证所提模型的有效性，在配置 CPU 为 Intel(R)i7-6700@3.40 GHz，内存为 8.00 GB、Windows 版本为 Windows10 家庭中文版的计算机上进行了仿真实验。假设实验是在无背景噪声环境下进行的，以一个模拟累加计算器为目标程序，在程序第一轮累加开始处、第 10 轮累加结束处、第 20 轮累加结束处分别设置检查点 e_0、e_1、e_2，即在 e_0、e_1、e_2 处置入场景信息监测器。

在检查点 e_0 与 e_1 之间随机注入干扰程序 t1.start()，而 t1.start()是检查点 e_0 与 e_1 之间的代码，使累加器重复运行了检查点 e_0 与 e_1 之间的代码，会引起程序的系统状态发生变化，例如，程序运行到检查点 e_2 的时间变长；在检查点 e_0 与 e_1 之间随机注入干扰程序 t1.join()，而 t1.join()直接调用检查点 e_1 与 e_2 之间的代码，使累加器绕过检查点 e_1，运行了检查点 e_1 与 e_2 之间的代码，会引起程序的系统状态发生变化，例如，程序运行到检查点 e_2 时间变短；在检查点 e_1 与 e_2 之间随机注入干扰程序 t1.start()，使累加器重复运行了检查点 e_0 与 e_1 之间的代码，会引起程序的系统状态发生变化，例如，程序运行到检查点 e_2 的时间变长；在检查点 e_1 与 e_2 之间随机注入干扰程序 t1.join()，而 t1.join()直接调用检查点

e_1 与 e_2 之间的代码，使累加器重复运行了检查点 e_1 与 e_2 之间的代码，会引起程序的系统状态发生变化，例如，程序运行到检查点 e_2 的时间变长。通过这 4 种随机注入干扰程序的方式模拟目标程序异常运行。在收集数据时，通过监测器捕获了目标程序以及在检查点 e_0、e_1 与 e_2 之间随机注入干扰程序 t1.start()或 t1.join()这两类程序后并行运行时检查点 e_0、e_1、e_2 的场景信息。

通过多次运行以上 2 类程序，直到收集了检查点 e_1 的状态转移到检查点 e_2 任一状态的所有情况，共收集了 8 794 条数据，每条数据包含检查点场景信息的各个指标。对于收集的数据中重复的场景信息则没有实验参考价值，所以舍弃不合理的数据后，作为实验的数据总共是 8 082 条。将收集的正常场景信息和异常场景信息输入 RBF 神经网络进行训练，对于处于两者之间的判定为临界场景信息，根据训练好的 RBF 神经网络对每个检查点的数据集进行分类判别。将收集的场景信息分为正常、临界、异常状态的样本，然后利用分类好的样本训练半加权马尔可夫模型的参数。RBF 神经网络和半加权马尔可夫模型建立之后，就利用监测器对运行程序进行监测，根据检查点 e_1 的状态预测检查点 e_2 的状态。

5.1 RBF 神经网络的建立

RBF 神经网络在 Matlab R2015a 平台上建立，网络采用 600 条记录，按照 70%、15%、15%的比例分为训练样本集、校正样本集、测试样本集，预设精度为 0.001，学习率设置为 0.01。

为了排除由于场景信息的各个指标因数量级或量纲上的差别而造成的影响，可用以下计算式进行归一化。

$$\text{新数据} = \frac{\text{采集数据} - \min}{\max - \min}$$

其中，max 和 min 分别表示各指标中的最大值和最小值。

在训练 RBF 神经网络时，将检查点的场景信息作为输入，检查点状态作为输出，所以输入层神经元数量为 6 个，输出层神经元数量为 3 个，设 C_1、C_2、C_3 分别表示检查点正常状态、临界状态、异常状态，将这 3 个状态的输出形式分别设为[1,0,0]、[0,1,0]和[0,0,1]。若训练后应用 RBF 神经网络分类处理输出为[1,0,0]，则相关样本属于正常类；若 RBF 神经网络分类处理输出为[0,1,0]，则相关样本属于临界类；若 RBF 网络分类处理输出为[0,0,1]，则相关样本属于异常类。

根据神经网络的运行机制分析可知[20]，影响神经网络泛化能力的因素可以分成两类：一类是样本的影响，另一类是来自神经网络本身的影响。本实验主要分析 RBF 神经网络本身对其泛化能力的影响，即通过分析连接权值及训练精度对泛化能力的影响，从而确定 RBF 神经网络的结构。

下面，通过训练数据集、校正数据集对 RBF 神经网络的连接权值及训练精度进行确定，并以网络的平均输出误差为评判指标，然后测试 RBF 神经网络的预测能力，输入测试样本，以分类正确率为评判指标。初始设置隐含层神经元数为 10 个，并每次以 10 个训练样本进行迭代训练 RBF 神经网络，而且隐含层神经元个数随着样本的增加而增加，同时，通过式(2)得出网络在每次迭代训练的权值，通过式(4)得出网络的平均输出误差，直到每一个训练样本都曾被用于隐含层神经元中心，然后通过校正样本集增加隐含层神经元个数，如果此时网络的平均输出误差比训练的误差小，则校正网络的权值及平均输出误差；随着隐含层神经元个数的增加，如果此时网络的平均输出误差比前一次增加的隐含层神经元校正的误差大，则保持前一次增加的隐含层神经元个数，说明网络的平均输出误差已达到最小值，此时 RBF 神经网络结构建立完成。然后通过测试数据集测试网络的预测能力，检查点 e_1 与检查点 e_2 的 RBF 网络训练和校正的平均输出误差结果分别如图 4 和图 5 所示，测试分类的预测精度分别如表 2 和表 3 所示。

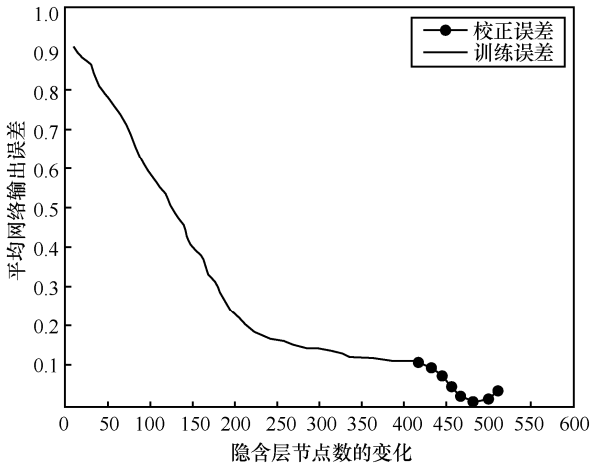


图 4 检查点 e_1 的 RBF 网络输出误差变化曲线

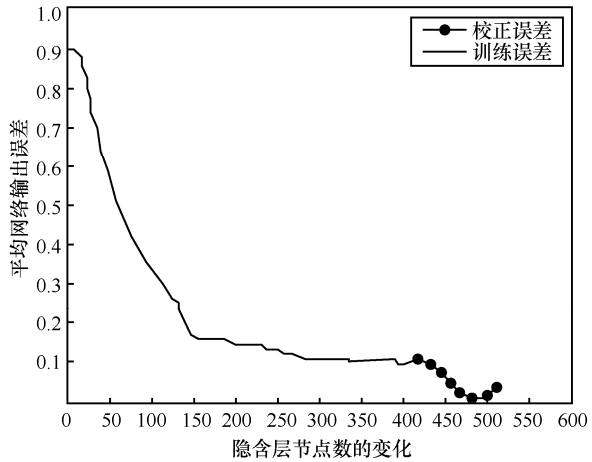


图 5　检查点 e_2 的 RBF 网络输出误差变化曲线

表 2　　检查点 e_1 的测试分类矩阵

类别	C_1	C_2	C_3	总计	正确率
C_1	90	0	0	90	100%
C_2	0	86	4	90	95.56%
C_3	0	7	83	90	92.22%
总计	90	93	87	270	95.93%

表 3　　检查点 e_2 的测试分类矩阵

类别	C_1	C_2	C_3	总计	正确率
C_1	90	0	0	90	100%
C_2	0	82	8	90	91.11%
C_3	0	6	84	90	93.33%
总计	90	88	92	270	94.81%

注：沿行方向是期望类别结果，沿列方向是实际类别结果。

从图 4 和图 5 可以看出，随着隐含层神经元个数的增加，RBF 神经网络的平均输出误差降低，当输出误差达到最小时，继续增加隐含层神经元个数，输出误差会增大，删除使输出误差增大的多余隐含层神经元个数，确定检查点 e_1 的 RBF 网络的隐含层神经元数为 480 个，检查点 e_2 的 RBF 网络的隐含层神经元数为 470 个，所以此时检查点 e_1 与 e_2 的 RBF 分类系统就建立起来了。

表 2 和表 3 的状态分类矩阵实验结果表明，采用 RBF 神经网络能够完成对检查点 e_1 与 e_2 状态的正确分类，总的正确率分别达到 95.93%和 94.81%。因此可以确定此时 RBF 网络的预测具有很好的泛化能力。

5.2　半加权马尔可夫模型的建立

在计算场景信息的权重时，采用的是 0.1～0.9 标度，如表 4 所示。因为 0.1～0.9 标度相对其他的标度来说，标度的均匀性和标度的可感知性是最好的[21]。

表 4　　0.1~0.9 标度取值

标度	取值
Ⅰ	0.5
Ⅱ	0.6
Ⅲ	0.7
Ⅳ	0.8
Ⅴ	0.9
Ⅵ	0.1, 0.2, 0.3, 0.4

采用 FAHP 方法计算场景信息 g_{T}、$g_{\Delta\mathrm{T}}$、g_{P}、$g_{\Delta\mathrm{P}}$、g_{M}、g_{B} 对应的权重，构建优先模糊矩阵 $\boldsymbol{A}$ 为

$$\boldsymbol{A}=\begin{bmatrix}0.5 & 0.4 & 0.3 & 0.4 & 0.6 & 0.6\\ 0.6 & 0.5 & 0.3 & 0.4 & 0.6 & 0.4\\ 0.7 & 0.7 & 0.5 & 0.6 & 0.6 & 0.7\\ 0.6 & 0.6 & 0.4 & 0.5 & 0.6 & 0.7\\ 0.4 & 0.4 & 0.4 & 0.4 & 0.5 & 0.6\\ 0.4 & 0.6 & 0.3 & 0.3 & 0.4 & 0.5\end{bmatrix}$$

根据式(6)计算得到 $(\omega_1,\omega_2,\omega_3,\omega_4,\omega_5,\omega_6)=(0.16, 0.16, 0.19, 0.18, 0.16, 0.15)$。

根据式(5)计算得到检查点 e_1 的正常、临界的平均场景值 $(\overline{S_1},\overline{S_2})=(33.94, 37.07)$。

当检查点 e_1 正常时，根据式(5)计算得到检查点 e_2 的正常状态、临界状态、异常状态的平均场景值 $(\overline{S_{11}},\overline{S_{12}},\overline{S_{13}})=(51.73, 58.86, 68.53)$。

当检查点 e_1 临界时，根据式(5)计算得到检查点 e_2 的正常状态、临界状态、异常状态的平均场景值 $(\overline{S_{21}},\overline{S_{22}},\overline{S_{23}})=(56.58, 62.06, 69.51)$。

比较 $\dfrac{\overline{S_1}}{S_{1n}}$ 与 $\dfrac{\overline{S_2}}{S_{2n}}$ 的大小，根据式(6)计算得到 $(\mu_1,\mu_2,\mu_3,\mu_4,\mu_5,\mu_6)=(0.500\,8, 0.499\,2, 0.495\,6, 0.504\,4, 0.490\,8, 0.509\,2)$。

根据收集的正常程序运行和加入干扰程序运行的数据，对这两类数据利用 RBF 进行分类判别，统计得到检查点 e_1 正常和临界状态在数据收集结束时的数据量分别是 6 660 和 1 422，所以检查点 e_1 的状态概率向量 $\boldsymbol{y}_i'=(0.824, 0.176)$，未加权转移概率矩阵为

$$\boldsymbol{P}=\begin{bmatrix}0.184 & 0.453 & 0.363\\ 0.114 & 0.333 & 0.553\\ 0.500 & 0.500 & 0\end{bmatrix}$$

根据式(7)对未加权转移概率进行加权 μ，则半加权马尔可夫模型的转移概率矩阵 $\boldsymbol{P}'$ 为

$$\boldsymbol{P}'=\begin{bmatrix}0.186 & 0.452 & 0.362\\ 0.112 & 0.332 & 0.556\\ 0.500 & 0.500 & 0\end{bmatrix}$$

通过以上步骤，建立初始半加权马尔可夫模型。

5.3 半加权马尔可夫模型的预测

程序在检查点 e_1 与 e_2 之间运行时间的增量最短是 27.353 s，而通过检查点 e_1 状态预测检查点 e_2 状态的时间是 16.239 s，这表明利用半加权马尔可夫模型进行预测是有意义的。

本实验选择 50 个程序周期对检查点 e_2 的状态进行评估，第 1、5、8、11、15、17、23、30、37、41、44、48 周期在检查点 e_0 与 e_1 之间注入干扰程序 t1.start()，并行运行目标程序和注入干扰后的程序；第 4、7、20、22、33、39 周期使累加器正常运行；第 14、23、30、34、35、36、38、40、43 周期在检查点 e_1 与 e_2 之间注入干扰程序 t1.join()，并行运行目标程序和注入干扰后的程序，收集这 3 种情况下的检查点 e_1、e_2 的场景信息进行实验。检查点 e_2 状态的散点评估如图 6 所示。

本文所提模型 CBSI-TM 对这 50 个周期的检查点 e_2 状态进行如下预测：第 1 个周期的检查点 e_1 场景值为 34.57，计算 $\left(\frac{S_1}{S_{11}},\frac{S_1}{S_{12}},\frac{S_1}{S_{13}}\right)=(0.668\,3,0.587\,3,0.504\,5)$，然后通过式(6)计算权重 $(\mu_1,\mu_2,\mu_3,\mu_4,\mu_5,\mu_6)=(0.502\,5,0.497\,5,0.497\,9,0.502\,1,0.493\,1,0.506\,9)$，对矩阵 $\boldsymbol{P}$ 的前两行重新加权，从而得到新的半加权马尔可夫转移概率矩阵为

$$\boldsymbol{P}'=\begin{bmatrix}0.186 & 0.454 & 0.360\\ 0.112 & 0.332 & 0.556\\ 0.500 & 0.500 & 0\end{bmatrix}$$

令 $\boldsymbol{p}'=\begin{bmatrix}0.186 & 0.454 & 0.360\\ 0.112 & 0.332 & 0.556\end{bmatrix}$，然后根据 $\boldsymbol{y}_{i+1}=\boldsymbol{y}_i'\boldsymbol{p}'$ 计算得到检查点 e_2 的状态概率向量 $\boldsymbol{y}_{i+1}=(0.173,0.433,0.394)$，最大的概率为 0.433，其对应检查点 e_2 的状态 2，即临界状态，与图 6 中实际状态一样。对第 2~50 个周期的检查点 e_2 的状态也按照以上步骤来计算。

将本文所提模型 CBSI-TM 与文献[3]的基于软件行为的检查点风险评估信任模型 CBRA-TM 和 RBF 神经网络分类方法进行比较，对这 50 个周期检查点 e_2 的状态进行评估，其结果如图 6 所示。程序运行到检查点 e_1 的时间 t_i 和检查点 e_2 的时间 t_{i+1} 与各模型评估检查点 e_2 状态所需的时间 Δt 的总时间开销 $t=t_i+\Delta t$ 或 $t=t_{i+1}+\Delta t$ 如图 7 所示。图 6 显示了与实际检查点 e_2 的状态相比，各模型对检查点 e_2 状态评估的结果。从图 6 可以得到以下结论。

1) 本文 CBSI-TM 模型只是在第 2、29、50 周期对检查点 e_2 状态的预测稍有偏差，因为在计算加权转移概率的权值时，除了检查点 e_1 是软件当前运行的场景值，检查点 e_2 所有状态的场景值都是均值，与实际稍有偏差，导致预测检查点 e_2 的状态有偏差，所以其正确率为 94%，而且 CBSI-TM 可以对检查点 e_2 的状态进行双重评估，从 5.1 节测试得出，RBF 分类方法对检查点 e_2 状态评估的平均正确率为 94.81%，所以 CBSI-TM 进行双重评估的平均正确率为 94.405%。

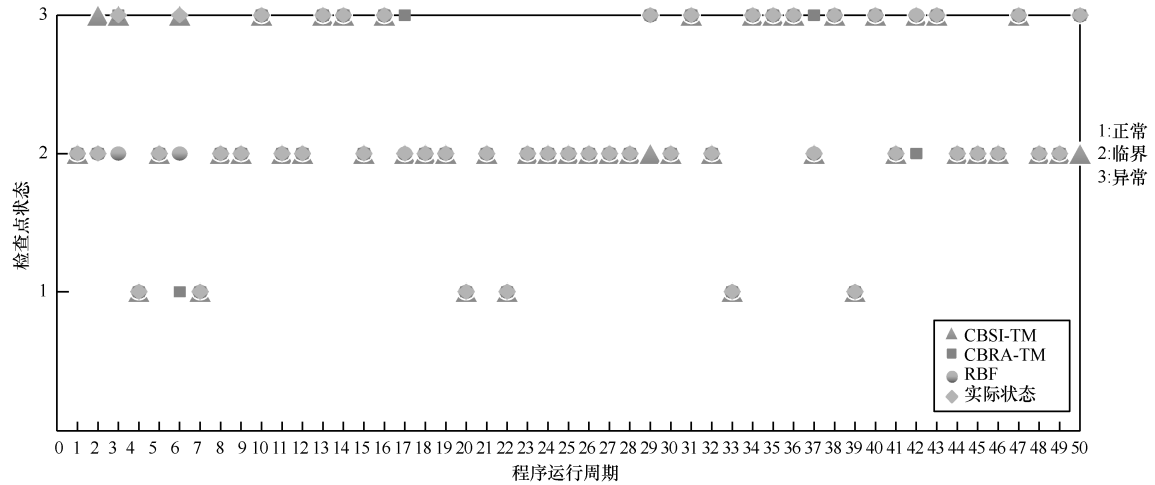


图 6 评估检查点 e_2 状态的散点

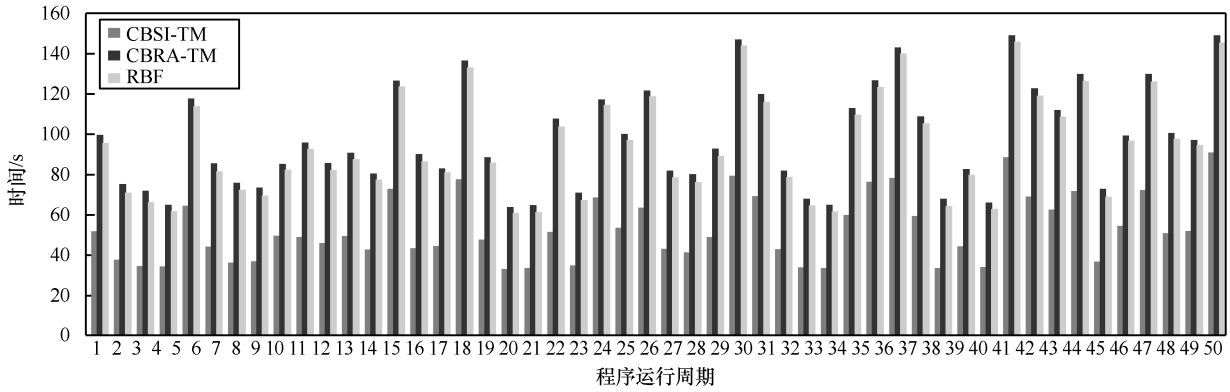


图 7　评估检查点 e_2 状态总时间开销的柱状图

2) 当采用 CBRA-TM 模型评估检查点 e_2 的状态时，在第 6、17、37、42 周期对检查点 e_2 状态的评估与实际状态有所不同，因为通过检查点发生风险的可能性与设定的阈值对比的方式来判断检查点的状态时，阈值的设定会导致对检查点状态的评估有偏差，所以其正确率为 92%。

3) 当采用 RBF 神经网络分类方法时，在第 3、6 周期对检查点 e_2 状态的评估与实际状态有所不同，因此其正确率为 96%。

4) CBSI-TM 模型评估的正确率低于 RBF 分类方法，这可能是因为在检查点 e_1、e_2 之间存在干扰，在检查点 e_1 预测时引起检查点 e_1 场景信息变化不明显，导致预测检查点 e_2 异常状态有偏差。

因为 CBSI-TM 模型可以在程序运行到检查点 e_1 时对检查点 e_2 的状态进行预测，而 CBRA-TM 模型和 RBF 神经网络分类方法只有在程序运行到检查点 e_2 时，才对检查点 e_2 的状态进行评估，所以从图 7 可以得到以下结论。

1) CBSI-TM 模型评估检查点 e_2 状态的总时间开销比 CBRA-TM 模型和 RBF 神经网络评估检查点 e_2 状态的总时间开销明显少，RBF 比 CBRA-TM 模型的时间开销少，这是因为 CBRA-TM 模型需要计算场景值并与其正常范围进行对比，才能评估检查点 e_2 的状态，而 RBF 通过将检查点 e_2 状态信息输入 RBF 神经网络即可得到检查点 e_2 的状态，因此处理检查点 e_2 信息的时间比 CBRA-TM 模型少，但 RBF 与 CBRA-TM 模型均是在程序运行到检查点 e_2 处，才对检查点 e_2 状态进行评估，所以总的时间开销是相似的。

2) 利用 CBSI-TM 模型对检查点 e_1 的状态进行评估后，预测检查点 e_2 的状态是合理和有效的，而且能实时调整半加权马尔可夫模型的参数，实现了动态预测。

3) 该实验也体现出 CBSI-TM 模型具有较强的针对性，只要在某个程序运行轨迹中设置检查点，发现检查点之间的状态变化规律，就能较为直观地反映软件未来运行趋势的可信情况，从而实现对软件的异常行为提前做出调控。

6　结束语

本文提出了一种基于检查点场景信息的软件行为可信预测模型，该模型通过 RBF 神经网络和半加权马尔可夫模型实现对检查点状态的可信评估方法，利用半加权马尔可夫模型有效地预测下一个检查点的状态，并且可以在程序运行到下一个检查点时，利用 RBF 神经网络分类器评估下一个检查点的状态，实现了对下一个检查点状态双重评估的效果。实验表明，该模型能够较准确和合理地预测检查点的状态来判断软件未来运行趋势的可信情况。下一步的工作将对软件的行为轨迹进行分析，进一步提高评估软件可信情况的合理性和精确度。

参考文献：

[1] 万烂军, 李长云, 贺宗梅. 分布式软件的交互行为监测机制的研究[J]. 计算机工程与应用, 2011, 47(5): 60-64.
WAN L J, LI C Y, HE Z M. Research on interactive behavior monitoring mechanism of distributed software[J]. Computer Engineering and Applications, 2011, 47(5): 60-64.

[2] 李珍, 田俊峰, 常卓, 等. 基于检查点的分布式软件监控与可信性评价[J]. 通信学报, 2016, 7(3): 7-18.
LI Z, TIAN J F, CHANG Z, et al. Distributed software monitoring and trustworthiness evaluation based on checkpoints[J]. Journal on Communications, 2016, 37(3): 7-18.

[3] 刘玉玲, 杜瑞忠, 冯建磊, 等. 基于软件行为的检查点风险评估信

任模型[J]. 西安电子科技大学学报(自然科学版), 2012, 39(1): 179-190.
LIU Y L, DU R Z, FENG J L, et al. Trust model of software behaviors based on checkpoint risk assessment[J]. Journal of Xidian University, 2012, 39(1): 179-190.

[4] 吴佳, 曾惟如, 陈瀚霖, 等. 基于隐马尔可夫模型的软件状态评估预测方法[J]. 软件学报, 2016, 27(12): 3208-3222.
WU J, ZENG W R, CHEN H L, et al. Approach of measuring and predicting software system state based on hidden Markov model[J]. Journal of Software, 2016, 27(12): 3208-3222.

[5] GUO X, DUTTA R G, JIN Y. Eliminating the hardware-software boundary: a proof-carrying approach for trust evaluation on computer systems[J]. IEEE Transactions on Information Forensics and Security, 2017, 12(2): 405-417.

[6] 王德鑫, 王青, 贺劼. 基于证据的软件过程可信度模型及评估方法[J]. 软件学报, 2017, 28(7): 1713-1731.
WANG D X, WANG Q, HE J. Software process reliability model based on evidence and evaluation method[J]. Journal of Software, 2017, 28(7): 1713-1731.

[7] 贾晓辉, 张文宁, 刘安战. 分级的软件可信评估模型研究及应用[J]. 计算机科学, 2017, 44(4): 169-172.
JIA X H, ZHANG W N, LIU A Z. Research and application of hierarchical software trustworthiness evaluation model[J]. Computer Science, 2017, 44(4): 169-172.

[8] 王犇, 周兴社, 杨亚磊. 基于多属性熵权合成的软件可信等级评估方法[J]. 微电子学与计算机, 2014, 31(6): 21-24.
WANG B, ZHOU X S, YANG Y L. Software trustworthiness grade evaluation method based on multiattribute entropy weight synthesis[J]. Microelectronics & Computer, 2014, 31(6): 21-24.

[9] LI K W, LIU L, ZHAI J N. Reliability evaluation model of component-based software based on complex network theory[J]. Quality and Reliability Engineering International, 2017, 33(3): 543-550.

[10] OKAMURA H, DOHI T. Towards comprehensive software reliability evaluation in open source software[C]. IEEE International Symposium on Software Reliability Engineering. 2015: 121-129.

[11] LIU L, JIANG Z. Research on software reliability evaluation technology based on BP neural network[C]//IEEE/ACIS International Conference on Computer and Information Science. 2016: 1-4.

[12] 匡芳君, 王艳华, 唐贤瑛. 基于 RBF 神经网络的客户分类模型[J]. 长沙理工大学学报, 2005, 2(4): 70-73.
KUANG F J, WANG Y H, TANG X Y. Customer classification model based on RBF neural network[J]. Journal of Changsha University of Science and Technology, 2005, 2(4): 70-73.

[13] 许智慧. 马尔可夫状态转移矩阵的求解方法研究[D]. 哈尔滨: 东北农业大学, 2013.
XU Z H. Study on the solution of Markov state transition matrix[D]. Harbin: Northeast Agricultural University, 2013.

[14] JØSANG A. A logic for uncertain probabilities[J]. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2001, 9(3): 279-311.

[15] 刘辉, 白峰衫. 基于混合高斯过程模型的高光谱图像分类算法[J]. 高校应用数学学报, 2010, 25(4): 379-385.
LIU H, BAI F S. Hyperspectral image classification based on mixed gaussian process model[J]. Applied Mathematics & A Journal of Chinese Universities, 2010, 25(4): 379-385.

[16] 王爱萍, 张功营, 刘方. EM 算法研究与应用[J]. 计算机技术与发展, 2009, 19(9): 108-110.
WANG A P, ZHANG G Y, LIU F. Research and application of EM algorithm[J]. Computer Technology and Development, 2009, 19(9): 108-110.

[17] 张雨浓, 李克讷, 谭宁. 中心、方差及权值直接确定的 RBF 神经网络分类器[J]. 计算机技术与自动化, 2009, 28(3): 5-9.
ZHANG Y N, LI K N, TAN N. RBF neural network classifier with center, variance and weight directly determined[J]. Computer Technology and Automation, 2009, 28(3): 5-9.

[18] 张雨浓, 王茹, 廖柏林, 等. 权值与结构双确定法的 RBF 神经网络分类器[J]. 计算机技术与自动化, 2014, 33(3): 1-7.
ZHANG Y N, WANG R, LIAO B L, et al. Both weights and structure determination method of RBF neural network classifier[J]. Computer Technology and Automation, 2014, 33(3): 1-7.

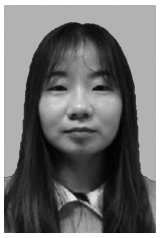
[19] 张吉军. 模糊层次分析法[FAHP][J]. 模糊系统与数学, 2000, 14(2): 80-88.
ZHANG J J. Fuzzy analytical hierarchy process[J]. Fuzzy Systems and Mathematics, 2000, 14(2): 80-88.

[20] 周文冬. 基于 PSO 的自组织 RBF 神经网络优化设计及应用研究[D]. 北京: 北京工业大学, 2016.
ZHOU W D. Optimazation and applizaction of self-organizing network based on PSO[D]. Beijing: Beijing University of Technology, 2016.

[21] 骆正清, 杨善林. 层次分析法中几种标度的比较[J]. 系统工程理论与实践, 2004(9): 51-60.
LUO Z Q, YANG S L. Comparative study on several scales in AHP[J]. Systems Engineering Theory and Practice, 2004(9): 51-60.

[作者简介]

田俊峰（1965−），男，河北保定人，河北大学教授、博士生导师，主要研究方向为信息安全与分布式计算。



郭玉慧（1992−），女，山西朔州人，河北大学硕士生，主要研究方向为信息安全与分布式计算。