

# 软件老化与再生研究综述

王艳超,姚江毅,李雄伟\*

(陆军工程大学石家庄校区,河北 石家庄 050003)

**摘要:**软件老化已经成为造成软件可靠性降低的主要因素。为了缓解软件老化对软件的不利影响,研究人员提出了软件再生技术,在软件性能恶化到一定程度时主动停止正在运行的服务或程序,清理内部状态,释放系统资源,使软件性能恢复到比较健康的状态,预防软件发生意外的崩溃。软件再生技术已经成为应对软件老化的方法,自提出以来研究人员围绕老化状态识别预测与提高再生效益展开研究。随着软件复杂度的提高和预测、监测等理论技术的发展,越来越多研究人员综合运用状态预测方法和再生策略研究方法来提高预测精度和再生效益。该文梳理总结了软件老化与再生的相关研究,首先根据研究的老化参数的不同将软件老化划分为资源消耗型和性能退化型老化;然后重点分析了软件老化状态预测研究和再生策略研究方法,并对软件再生实施技术进行分析梳理;最后,在总结近期研究特点的基础上,对下一步工作提出了几点展望。

**关键词:**软件老化;软件再生;老化预测;再生策略;再生成本

**中图分类号:**TP311.5

**文献标识码:**A

**文章编号:**1673-629X(2022)10-0001-06

**doi:**10.3969/j.issn.1673-629X.2022.10.001

## Review of Software System Aging and Rejuvenation

WANG Yan-chao, YAO Jiang-yi, LI Xiong-wei\*

(Shijiazhuang Campus, Army Engineering University of PLA, Shijiazhuang 050003, China)

**Abstract:** Software aging has become the main factor of the deterioration of software reliability. In order to alleviate the negative impact of software aging for software, researchers present software rejuvenation technology. When the software performance deteriorates to a certain extent, the running services or programs are stopped to clean up the internal status and release system resources to restore the software performance to a healthy state and prevent unexpected software crashes. Software rejuvenation technology has become a method to deal with software aging. Since it was proposed, researchers have focused on the identification and prediction of aging state and improving the efficiency of rejuvenation. With the increase of software complexity and the development of prediction and monitoring theory and technology, more and more researchers comprehensively take the method of predicting state and the strategy of rejuvenation to improve the accuracy of prediction and the efficiency of rejuvenation. We summarize the related research on software aging and rejuvenation. Firstly, software aging is divided into resource consumption type and performance degradation type according to different aging parameters. Then the research methods of software aging state prediction and rejuvenation strategy are analyzed, and the implementation technology of software rejuvenation is analyzed and combed. Finally, based on the summary of recent research characteristics, some prospects for the future research directions are pointed out.

**Key words:** software aging; software rejuvenation; aging prediction; rejuvenation strategy; rejuvenation cost

## 0 引言

软件老化是在一个长时间持续运行的软件中出现的系统资源泄漏、数据腐烂等,使状态退化最终导致软件崩溃的现象,其根本原因是软件中存在老化相关的缺陷(Aging-Related bugs)<sup>[1]</sup>。在二十世纪六十年代,研究人员就发现军用软件系统在运行过程中出现故障率增加、性能持续下降的现象。随着软件复杂度提高

和应用范围的扩展,研究人员在越来越多的软件中发现老化现象。软件老化会造成系统可靠性降低,在一些关键系统中甚至会造成巨大人员伤亡<sup>[2]</sup>。

为了缓解软件老化的影响,Huang等人<sup>[3]</sup>提出软件再生技术,即有计划地暂停正在运行的软件,清理内部状态,使软件恢复到较为健康的状态。软件再生是一种预反应式的主动抗衰技术,再生期间系统不能提

收稿日期:2021-11-15

修回日期:2022-03-17

基金项目:国家自然科学基金(62071483)

作者简介:王艳超(1992-),男,硕士,研究方向为军用软件安全与保障;通信作者:李雄伟(1975-),男,博士,教授,研究方向为信息系统与安全。

供服务,一定程度上降低系统的可用性。因此在软件再生技术被提出以来,众多研究人员致力于设计优化再生策略,降低再生成本,最大化软件服务可用性。

## 1 软件老化分类

在软件老化研究中主要通过老化参数的变化来识别软件老化状态。根据老化参数类型可以将软件老化划分为资源消耗型老化和性能退化型老化。

### 1.1 资源消耗型老化

资源消耗型老化是指由于系统内部错误使得内存、线程、文件锁等系统资源被逐渐消耗,可用资源逐渐减少的老化现象。Garg 等人<sup>[4]</sup>指出内存泄漏会使软件出现性能下降、挂起或崩溃的老化现象,之后大多数关于软件系统老化预测的研究都将内存资源等作为老化模型的输入进行老化分析。徐萍<sup>[5]</sup>通过跟踪系统的运行状况,对系统内存、进程和 CPU 占用情况进行采样分析。Cassidy 等人<sup>[6]</sup>发现数据库管理系统中的共享内存池锁资源的争用会严重导致系统性能下降。Dony 等人<sup>[7]</sup>利用套接字描述符资源研究 Java 中的异常处理。负载的强度也会影响系统资源的消耗趋势,Matias 等人<sup>[8]</sup>设计加速老化试验证明负载对系统资源消耗的影响。

### 1.2 性能退化型老化

性能退化是软件老化最直观的表现,包括响应时间变长、吞吐量和处理速度下降等,是软件系统状态退化的综合反映。Levitin 等人<sup>[9]</sup>根据系统处理速度的变化,结合系统工作量、可用内存构建了基于事件转移的老化状态模型。Zhao 等人<sup>[10]</sup>以响应时间作为老化指标检测 Apache HTTP 服务器老化趋势。不同的软件响应时间不同,不能单纯以响应时间的长短定义软件性能的好坏,需要根据软件特点及用户的感受来划分响应时间的退化标准。

在安全关键软件系统存在由运算精度不足引起的老化,不属于上述两种类型。其主要原因是系统缺陷以及系统运算精度导致的数值错误的积累,目前还没有相应的老化参数来检测这种软件老化,只有其表现出来才会被发现,如美军爱国者导弹防御系统的数值误差导致导弹偏离目标<sup>[11]</sup>。

## 2 软件老化预测研究

软件老化预测就是在收集软件老化相关参数数据的基础上,运用数据挖掘提取数据特征,预测软件老化趋势,为制定合适的再生策略提供支持。老化预测的研究方法主要有时序分析法、机器学习等方法。

### 2.1 时序分析法

时序分析法是建立时序模型分析预测老化数据的

变化趋势。常见的时序模型有多元线性回归、回归平滑、自回归模型等。Li 等人<sup>[12]</sup>通过收集到的 Web 服务器的响应时间、可用内存、消耗的交换空间等参数数据构建了 ARMA/ARX 模型来预测 Web 服务器系统在人工负载下的资源消耗趋势。Grottke 等人<sup>[13]</sup>收集了在 Linux 系统环境下运行的 Apache Web 服务器在可控负载下资源使用情况,考虑数据的季节性,使用 AR 模型预测服务器资源消耗情况。Magalhaes 和 Silva<sup>[14]</sup>针对动态性强和不确定性的工作负载对 Web 服务器系统资源的影响,采用 ARIMA 和 Holt-Winters 模型估计应用程序和系统的老化相关参数,可以较好地模拟数据变化趋势及季节性特征。Araujo<sup>[15]</sup>分别使用线性模型、二次线性模型、指数增长模型和 S 曲线趋势模型预测 Eucalyptus 云计算框架下的内存消耗趋势,通过比较四种模型预测结果偏差和均方差评估模型的预测效果,其中二次趋势模型预测结果更准确。

### 2.2 机器学习

机器学习是一种复杂的数学分析方法,采用人工智能算法提取、拟合数据中的非线性特征,预测老化趋势。Andrzejak 等人<sup>[16]</sup>将机器学习算法(决策树、贝叶斯方法和支持向量机)用于主动检测系统老化,该方法对引起老化的瞬态故障敏感性较强。Alonso 等人<sup>[17]</sup>提出基于机器学习 MSP 算法对软件老化进行预测,该方法可以根据不同的系统参数自动构建预测模型,较好地适应各种复杂的老化场景。Zhao 等人<sup>[18]</sup>和 Jia 等人<sup>[19]</sup>利用 BP 神经网络搭建软件老化评估模型,利用 BP 神经网络的非线性映射和柔性网络结构,识别由于多种系统资源相互作用导致的软件老化的综合效应。BP 神经网络存在收敛速度慢和容易陷入局部极值的缺陷,在文献[20-22]分别利用人工蜂群算法、自适应遗传退火算法和粒子群退火算法对 BP 神经网络进行优化,提高了预测的准确度。随着深度神经网络模型的成熟,部分研究人员尝试利用深度神经网络来解决上述问题,党等人<sup>[23]</sup>使用 LSTM 模型预测 Web 软件系统的实时剩余寿命,取得一些成功。

同时,由于软件老化参数数据中既有线性特征,也有非线性特征,时序分析法可以很好地拟合线性特征,机器学习方法可以较好地拟合非线性特征,将两种方法结合起来可以提高老化预测的准确性。Yan<sup>[24]</sup>使用 ARIMA 模型表示系统资源消耗的线性趋势,同时使用 BP 人工神经网络来拟合趋势中隐藏的非线性特征和残差线性特征来构建混合模型,预测系统资源消耗趋势。试验表明,混合模型的预测结果要好于其他两种模型单独的预测结果。Liu 等人<sup>[25]</sup>提出混合老化预测模型 CSSAP,该模型集成了 ARIMA 模型和循环神经网络中的长短时记忆(Long Short-Term Memory,

LSTM)模型,能够很好地拟合计算资源使用数据时间序列的线性模式和挖掘非线性关系。

时序分析法算法简单,处理序列速度快,针对较稳定序列和有规律(如具有季节性、周期性)的序列预测精度高,但对于目前大多数结构复杂和逻辑复杂的软件系统的老化参数序列其预测效果不理想;机器学习的方法可以识别多种因素综合导致的老化特征,BP 人工神经网络和循环神经网络的预测精度都很高,但 BP 网络存在收敛速度慢、易陷入局部极值的问题,循环神经网络也存在梯度不稳、占用内存大等问题。综合运用时序分析法和机器学习方法结合了两者的优点,在预测精度上较单一的方法都有所提升。

### 3 软件再生策略研究

软件再生策略研究的目的是通过合适的再生计划,降低再生成本,最大化软件可用性。目前,再生策略研究方法主要有基于模型分析的方法、基于阈值的方法等。

#### 3.1 基于模型分析的方法

模型分析的方法是将软件状态和行为抽象为符合时间分布的数学状态概率模型来研究最优的再生策略<sup>[26]</sup>。常见的分析模型有马尔可夫链及其扩展模型、随机 Petri 网模型等。

##### (1) 马尔可夫链及其扩展模型。

马尔可夫链是一个离散随机过程,随机系统在每一时刻上的状态仅取决于前一时刻的状态。文献[3]将系统划分为包含鲁棒状态、易于失效状态、再生状态和失效状态的 4 种再生概率状态转移模型(见图 1)。

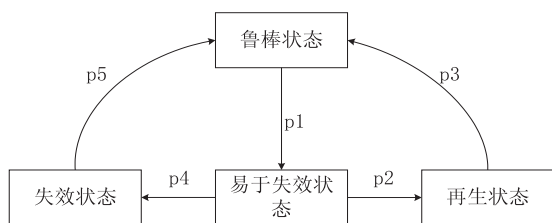


图 1 软件再生概率转移状态模型

在此基础上利用连续时间马尔可夫链(CTMC)对软件老化过程进行建模,通过状态转移概率矩阵预估再生停机成本。Xie 等人<sup>[27]</sup>对可用状态进一步扩展,将系统划分为 7 个状态,利用半马尔可夫模型对系统状态进行建模,实现了服务级和全系统级的两级再生策略,结合了服务级再生的时间优势和系统级再生的效果优势,进一步提高了再生效益。Chang 等人<sup>[28]</sup>构建了半马尔可夫模型,推导在云计算虚拟机监控器和虚拟机老化再生过程中的程序的服务可用性和任务完成的时间,通过实验得到再生效益最大的最佳再生间隔。文献[29]利用马尔可夫模型对嵌入式集群系统

及各节点系统老化进行整体建模分析,获得集群系统的可用性、平均故障时间、停机成本等。

##### (2) Petri 网模型。

Petri 网模型在描述系统并行性、分布性和不确定性等方面有较好的优势<sup>[30]</sup>。Garg 等人<sup>[31]</sup>使用马尔可夫随机再生 Petri 网模型对 C/S 类型系统行为建模求解系统稳态条件,评估最佳的再生时间间隔。Wang 等人<sup>[32]</sup>针对非指数分布,在马尔可夫再生过程(MRGP)基础上提出确定性和随机 Petri 网(Deterministic and Stochastic Petri Net, DSPN)模型对具有多个节点的周期性负载的集群系统进行老化建模研究,针对不同工作负载设计了标准再生、延迟再生和混合再生三种基于时间的再生策略。标准再生即每隔固定的时间段  $T$  后启动再生程序,延迟再生是负载高峰期到达再生时刻时等待负载降低后启动再生,而混合再生遇到负载高峰期时比较立即再生和延迟再生对性能的影响决定再生的执行。Zheng 等人<sup>[33]</sup>提出一种复合随机 Petri 奖励网建模一个经历非周期的再生和检查点恢复的软件系统,通过分析合适的再生触发时间范围,获得使系统稳态可用性最大的再生计划。

基于分析模型的方法是对软件系统的抽象,可以很容易的跨系统移植。由于把软件运行过程进行了简化假设,抽象为不同的状态过程,存在状态空间爆炸的问题,同时分析模型的方法只能提供渐进解或稳态解,在具体系统中效果并不精确。

#### 3.2 基于阈值的方法

基于阈值的再生策略是针对一些老化参数设定阈值,当监测到的参数值超过设定的阈值时,系统即触发再生程序。如图 2 所示,在  $T_1$ 、 $T_2$  监测到系统性能下降到阈值  $G_{th}$  以下,系统触发再生,使软件性能恢复到初始状态。

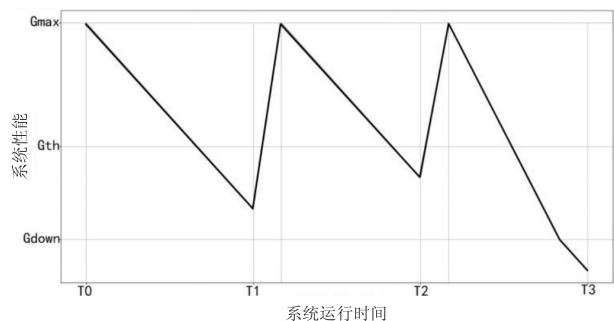


图 2 基于阈值的再生策略示意图

在基于阈值的方法中,阈值的选择需要丰富的经验知识以及了解所研究系统引起老化的指标有哪些,既要有效预防失效,又能避免无效的再生。Alonso 等人<sup>[34]</sup>研究网格服务系统老化再生时,采用虚拟中间件技术结合基于阈值的方法寻找最优再生机制。Meng 等人<sup>[35]</sup>在 J2EE 应用服务器软件再生策略研究中,采



用阈值的方法实现组件级、JVM 级和系统级三级再生。基于阈值的方法需要监测系统参数,频繁的监测会增加系统的负担,过大的监测间隔可能在两次监测之间发生故障或失效(图 2 中  $T_3$  监测点)。为了弥补这种缺陷,文献[17]提出使用预测老化的方法来优化监测间隔。在多软件系统中,不同的软件之间针对同一老化参数的大小和趋势是不同的,不能设置一个阈值来激活所有软件的再生,Sukhwani 等人<sup>[36]</sup>提出为每个软件的老化参数设置不同的再生阈值。

目前,部分研究人员将基于分析模型和阈值的方法结合起来,增强预测方法在具体软件系统中的灵活性。Ghobadi 和 Rashidi<sup>[37]</sup>将系统老化划分成 4 个等级,利用马尔可夫链构建分级老化再生模型,由可用内存阈值来触发再生,同时引入执行再生时间、实时系统服务量和再生操作数量为参数的成本函数来量化再生成本。Levitin 等人<sup>[9,38-40]</sup>基于事件转移的方法评估实时任务系统的性能以及执行实时任务的成功完成率,该方法与建模方法不同的是其不受故障时间和状态转换时间分布类型的限制,通过检测系统性能参数,利用再生决策算法决定再生的执行。并运用分级再生、周期检查点等技术进一步优化再生成本。

基于模型分析的方法不针对某一特定软件系统,迁移性好、节约研究时间,但只能求得系统可靠性的稳态解或渐进解,运用到某一具体软件系统时,再生效果并不理想,同时还存在状态空间爆炸问题;基于阈值的再生方法针对具体系统,需要统计分析大量系统运行数据,考虑系统运行具体特征,能够处理实时故障行为,再生策略实施效果好,但方法迁移性差。

## 4 软件再生实施技术

再生实施技术是触发再生后,以什么技术实施再生实现改善系统性能状态的目标。软件再生实施技术主要有重启、检查点恢复和系统迁移等。

### 4.1 重 启

重启即停止正在运行的系统、软件、进程或服务,清理内部状态、释放占用的系统资源,然后重新启动系统、软件、进程或服务,使系统性能恢复到一个较高的水平。根据重启的粒度从大到小有操作系统重启、虚拟机重启、应用软件重启、进程重启、服务重启。粗粒度重启中包含着细粒度重启,随着粒度的减小,重启时间成本和再生效果降低。在一些软件系统中,通过结合不同粒度的分级再生可以优化再生策略,但是在细粒度老化分析难度较大的软件中,可能会产生比较大的时间开销。

### 4.2 检查点恢复

检查点技术是周期或非周期地将内存中数据保存

到辅助存储器中,系统执行再生后将最近一次保存的副本恢复到内存中,可以节省再生后重复执行的时间<sup>[41]</sup>,常见的检查点技术包括增量检查、审计跟踪和备份恢复技术<sup>[42]</sup>。增量检查只将修改过的内存页面保存到检查点文件;审计跟踪技术记录了系统的一系列操作,能够恢复到系统崩溃前的正确状态,也能取消事务;备份恢复技术保留一个包含当前值和一个包含以前值的备份版本,当系统崩溃时,使用备份版本恢复到崩溃前的正确状态。

### 4.3 系统迁移

系统迁移是在虚拟化系统中,当正在处理业务的虚拟机发生故障时,使用同一主机上的备用虚拟机或将备用虚拟机迁移到另一台主机上继续提供服务的技术。在云计算快速发展过程中,虚拟化技术发挥了重要作用,相关软件老化再生研究也越来越多。在虚拟机监视器和虚拟机老化再生研究中,普遍运用了系统迁移再生技术。在文献[43-44]中运用了故障转换和系统迁移(包括实时虚拟机动态迁移技术)来缓解系统老化影响。

## 5 结束语

随着软件开发应用和数据挖掘理论的发展,软件老化与再生研究不断丰富。该文主要从软件老化分类、软件老化预测、软件再生策略研究和实施再生技术四个方面对软件老化和再生研究进行了分析梳理,未来,相关研究将呈现出一些新趋势:①随着软件复杂度的提高,各种老化因素之间的影响更加复杂,在软件运行过程中内存消耗与软件性能不一定是正相关。因此在今后的研究中需要考虑多种老化参数的综合作用,确保老化数据特征的完整性;②软件老化数据中含有线性和非线性特征,越来越多的研究人员采用时序分析法与机器学习相结合的方法来提高预测精度。目前深度学习算法已经非常成熟,特征提取能力要优于 BP 算法,可以将深度学习算法应用于软件老化预测;③随着软件老化状态预测精度的提高,在再生策略研究中更多地运用多粒度分级再生,但忽略了不同粒度之间、同级粒度之间再生效益相互影响关系的研究,未来可以在这一方面加强研究,进一步提高整体再生效益。

该文对软件老化与再生研究进行了梳理,并就未来研究提出几点粗浅的建议,希望能够起到抛砖引玉的作用,并为本领域研究提供一些启示。

### 参考文献:

- [1] GROTTKE M, TRIVEDI K S. Fighting bugs: remove, retry, replicate, and rejuvenate[J]. Computer, 2007, 40(2): 107-109.

- [2] SGOBBA T. B-737 MAX and the crash of the regulatory system[J]. *Journal of Space Safety Engineering*, 2019, 6(4):299-303.
- [3] HUANG Y, KINTALA C, KOLETTIS N, et al. Software rejuvenation: analysis, module and applications[C]//Twenty-fifth international symposium on fault-tolerant computing. Pasadena: IEEE, 1995:381-390.
- [4] GARG S, PULIAFITO A, TELEK M, et al. Analysis of preventive maintenance in transactions based software systems[J]. *IEEE Transactions on Computers*, 1998, 47(1):96-107.
- [5] 徐 萍. 基于 MRSPN 模型计算自恢复时间间隔[D]. 南京: 南京理工大学, 2005.
- [6] CASSIDY K J, GROSS K C, MALEKPOUR A. Advanced pattern recognition for detection of complex software aging phenomena in online transaction processing servers[C]//Proceedings of international conference on dependable systems and networks. Washington: IEEE, 2002:478-482.
- [7] DONY C, URTADO C, VAUTIER S. Exception handling and asynchronous active objects: issues and proposal[M]//Advanced topics in exception handling techniques. Berlin: Springer, 2006:81-100.
- [8] MATIAS R, BARBETTA P A, TRIVEDI K S, et al. Accelerated degradation tests applied to software aging experiments[J]. *IEEE Transactions on Reliability*, 2009, 59(1):102-114.
- [9] LEVITIN G, XING L, LUO L. Joint optimal checkpointing and rejuvenation policy for real-time computing tasks[J]. *Reliability Engineering & System Safety*, 2019, 182:63-72.
- [10] ZHAO J, TRIVEDI K S, GROTTKE M, et al. Ensuring the performance of Apache HTTP server affected by aging[J]. *IEEE Transactions on Dependable and Secure Computing*, 2013, 11(2):130-141.
- [11] GROTTKE M, NIKORA A P, TRIVEDI K S. An empirical investigation of fault types in space mission system software[C]//2010 IEEE/IFIP international conference on dependable systems & networks (DSN). Chicago: IEEE, 2010:447-456.
- [12] LI L, VAIDYANATHAN K, TRIVEDI K S. An approach for estimation of software aging in a web server[C]//Proceedings of international symposium on empirical software engineering. Nara: IEEE, 2002:91-100.
- [13] GROTTKE M, LI L, VAIDYANATHAN K, et al. Analysis of software aging in a web server[J]. *IEEE Transactions on Reliability*, 2006, 55(3):411-420.
- [14] MAGALHÃES J P, SILVA L M. Prediction of performance anomalies in web-applications based-on software aging scenarios[C]//2010 IEEE second international workshop on software aging and rejuvenation. San Jose: IEEE, 2010:1-7.
- [15] ARAUJO J, MATOS R, ALVES V, et al. Software aging in the eucalyptus cloud computing infrastructure: characterization and rejuvenation[J]. *ACM Journal on Emerging Technologies in Computing Systems*, 2014, 10(1):1-22.
- [16] ANDRZEJAK A, SILVA L. Using machine learning for non-intrusive modeling and prediction of software aging[C]//NOMS 2008-2008 IEEE network operations and management symposium. Salvador: IEEE, 2008:25-32.
- [17] ALONSO J, TORRES J, BERRAL J L, et al. Adaptive on-line software aging prediction based on machine learning[C]//2010 IEEE/IFIP international conference on dependable systems & networks (DSN). Chicago: IEEE, 2010:507-516.
- [18] ZHAO J, TRIVEDI K S, WANG Y B, et al. Evaluation of software performance affected by aging[C]//2010 IEEE second international workshop on software aging and rejuvenation. San Jose: IEEE, 2010:1-6.
- [19] JIA Y F, ZHOU Z Q, XUE K X, et al. Using neural networks to forecast available system resources: an approach and empirical investigation[J]. *International Journal of Software Engineering and Knowledge Engineering*, 2015, 25(4):781-802.
- [20] LIU J, MENG L. Integrating artificial bee colony algorithm and BP neural network for software aging prediction in IoT environment[J]. *IEEE Access*, 2019, 7:32941-32948.
- [21] 裴 瑞, 白尚旺, 党伟超, 等. 自适应遗传退火算法优化 BP 神经网络及其应用[J]. *计算机系统应用*, 2019, 28(7):109-113.
- [22] 王 荣, 白尚旺, 党伟超, 等. 粒子群退火算法优化的 BP 神经网络及其应用[J]. *计算机系统应用*, 2020, 29(1):244-249.
- [23] 党伟超, 李 涛, 白尚旺. 基于 LSTM 网络的 Web 软件系统实时剩余寿命预测[J]. *计算机系统应用*, 2021, 30(7):253-258.
- [24] YAN Y. Novel method to forecast software aging problems[J]. *The Journal of Engineering*, 2019, 2019(10):7237-7243.
- [25] LIU J, TAN X, WANG Y. CSSAP: software aging prediction for cloud services based on ARIMA-LSTM hybrid model[C]//2019 IEEE international conference on web services (ICWS). Milan: IEEE, 2019:283-290.
- [26] NING G, ZHAO J, LOU Y, et al. Optimization of two-granularity software rejuvenation policy based on the Markov regenerative process[J]. *IEEE Transactions on Reliability*, 2016, 65(4):1630-1646.
- [27] XIE W, HONG Y, TRIVEDI K. Analysis of a two-level software rejuvenation policy[J]. *Reliability Engineering & System Safety*, 2005, 87(1):13-22.
- [28] BAI J, CHANG X, MACHIDA F, et al. Analyzing software rejuvenation techniques in a virtualized system: service provider and user views[J]. *IEEE Access*, 2020, 8:6448-6459.
- [29] JAIN M, MANJULA T, GULATI T R. Software rejuvenation policies for cluster system[J]. *Proceedings of the National*

- Academy of Sciences, India Section A; Physical Sciences, 2016, 86(3):339–346.
- [30] WINSLETT M. Bruce Lindsay speaks out: on system R, benchmarking, life as an IBM fellow, the power of DBAs in the old days, why performance still matters, Heisenbugs, why he still writes code, singing pigs, and more[J]. ACM SIGMOD Record, 2005, 34(2):71–79.
- [31] GARG S, PULIAFITO A, TELEK M, et al. Analysis of software rejuvenation using Markov regenerative stochastic Petri net[C]//Proceedings of sixth international symposium on software reliability engineering. Toulouse: IEEE, 1995:180–187.
- [32] WANG D, XIE W, TRIVEDI K S. Performability analysis of clustered systems with rejuvenation under varying workload[J]. Performance Evaluation, 2007, 64(3):247–265.
- [33] ZHENG J, OKAMURA H, DOHI T. A phase expansion for non-Markovian availability models with time-based aperiodic rejuvenation and checkpointing[J]. Communications in Statistics–Theory and Methods, 2020, 49(15):3712–3729.
- [34] ALONSO J, SILVA L, ANDRZEJAK A, et al. High-available grid services through the use of virtualized clustering[C]//2007 8th IEEE/ACM international conference on grid computing. Austin: IEEE, 2007:34–41.
- [35] MENG H, HEI X, ZHANG J, et al. Software aging and rejuvenation in a j2ee application server[J]. Quality and Reliability Engineering International, 2016, 32(1):89–97.
- [36] SUKHWANI H, MATIAS R, TRIVEDI K S, et al. Monitoring and mitigating software aging on IBM cloud controller system[C]//2017 IEEE international symposium on software reliability engineering workshops (ISSREW). Toulouse: IEEE, 2017:266–272.
- [37] GHOBADI Z R, RASHIDI H. A software availability model based on multilevel software rejuvenation and markov chain[J]. Turkish Journal of Electrical Engineering & Computer Sciences, 2021, 29(2):730–744.
- [38] LEVITIN G, XING L, XIANG Y. Optimizing software rejuvenation policy for tasks with periodic inspections and time limitation[J]. Reliability Engineering & System Safety, 2020, 197:106776.
- [39] LEVITIN G, XING L, XIANG Y. Cost minimization of real-time mission for software systems with rejuvenation[J]. Reliability Engineering & System Safety, 2020, 193:106593.
- [40] LEVITIN G, XING L, HUANG H Z. Optimization of partial software rejuvenation policy[J]. Reliability Engineering & System Safety, 2019, 188:289–296.
- [41] ZHENG J, OKAMURA H, DOHI T. Availability analysis of software systems with rejuvenation and checkpointing[J]. Mathematics, 2021, 9(8):846.
- [42] TANTAWI A N, RUSCHITZKA M. Performance analysis of checkpointing strategies[J]. ACM Transactions on Computer Systems, 1984, 2(2):123–144.
- [43] CHANGA X, ZHANG Z, LI X, et al. Model-based survivability analysis of a virtualized system[C]//2016 IEEE 41st conference on local computer networks (LCN). Dubai: IEEE, 2016:611–614.
- [44] NGUYEN T A, MIN D, CHOI E, et al. Stochastic reward net-based modeling approach for availability quantification of data center systems[M]//Dependability engineering. Rijeka: InTech, 2018.