|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **序号** | **年份** | **作 者** | **文 章 题 目** | **针对的软件和致衰因素/表现.**  **评价指标选取** | **衰退检测/评估方法** | **预测未来状态 (剩余可用寿命)方法(analysis)** | **选择抗衰策略，实施抗衰技术** | **关 键 词** | **不 足** |
| **基于时间模型(系统建模，定时抗衰)** | | | | | | | | | |
|  | 1994 | David Lorge Parnas | Software Aging | 软件产品过时。内存泄露、线程未终止、存储空间碎片、数据溢出；操作(参数配置)不当 | 衰退不可阻止，是由需求变更引起的。在不断紧迫的升级、打补丁过程中，软件中遗留了一些errors。描述了软件衰退的起因、代价、预防软件退化的措施， | 预防措施：执行软件工程流程：良好的设计(灵活、预计可能的需求变化)、详尽的文档、软件评价、截肢(去掉频繁、草率的修改而陈旧的代码)、重构 |  |  | 主要针对软件产品功能的退化,草率的应对需求变更，引起代码质量下降；软件工程做的再好，仍不可避免会衰退 |
|  | 1995 | Yennun Huang, C. Kintala | [Software Rejuvenation：Analysis Module and Applications](../00-参考论文/(1995)%20Software%20Rejuvenation：Analysis%20Module%20and%20Applications（隐蔽性软件故障）.pdf) | 1. 电话计费系统： 2. 人工经验 | 无此步骤 | 连续马尔科夫链：建模、驻留时间和转移概率，解最优抗衰周期 | 考虑时间的、系统级停机、重启 | 不确定性因素，隐蔽性故障elusive bug，瞬时失效transient failure | 未考虑负载的影响；需要衰退先验知识 |
|  | 1995 | Garg. S | Analysis of software rejuvenation using Markov regenerative stochastic Petri net | 1. C/S软件系统的server 2. 失效率增加; 3. 人工经验 | 无此步骤 | 马尔科夫再生随机petri网：解最优抗衰周期,定期抗衰 | 考虑时间、和考虑时间&负载的;系统级停机、重启 | 早期基于度量的方法？ |  |
|  | 1996 | A. Pfening,  S. Garg,  A. Puliafito, K.S. Trivedi | Optimal Software Rejuvenation for Tolerating Soft Failures | 1. 电话通讯系统的server 2. 服务率下降 (事务丢失) 3. 人工经验 | 无此步骤 | 马尔科夫决策过程：性能下降到警戒值，无可接受;定合适抗衰时机：（look -ahead-n函数，最优最小成本函数） |  |  | 现有的大部分方法都缺乏固定的/一致的控制基础[] |
|  | 1997 | S. Garg,  A. Puliafito, M. Telek,  K. S. Trivedi | On the analysis of software rejuvenation policies | 1. 基于事务的系统 2. 失效率增加和服务率下降 3. 人工经验 | 无此步骤 |  | 两种策略：基于时间的，基于时间&负载的，评价方法：恢复后软件的稳定态可用性和事务的服务拒绝概率：系统级 | 软件的分析模型；bug瞬时性 | 假设条件较多；可用性如何评价？ |
|  | 1997 | 1. Avritzer, E.J. Weyuker | Monitoring Smoothly Degrading Systems for Increased qazDependability |  |  |  |  |  |  |
|  | 1998 | Garg. S | analysis of preventive maintenance in transactions based software systems | 1. 基于事务的软件系统的server(DBMS除外) 2. 服务率下降(丢失的事务数/概率)；挂起/崩溃失效率上升 3. 人工经验 | 无此步骤 | 马尔科夫再生模型：建模软件行为={t时刻软件的状态,t时刻队列容纳力,t}，求各状态驻留时间和转移概率，解最优抗衰周期。判断依据：稳定状态可用性、事务丢失的概率和事务预期响应时间的上限 | 两种策略：考虑时间,考虑时间&负载，评价指标：满足如下评价指标：软件可用性、丢失事务的概率、事务的响应时间；系统级 | Transient; does not recur upon re-execution;  time-based最完善的基于时间的方法。首次考虑了负载对衰退的影响 | 假设失效发生时，到来的事务会丢失；  模型精度的影响因子：考虑的衰退的影响因素 |
|  | 总结 | 基于时间模型(系统建模)（定时抗衰） | | 人工经验；  性能=丢失的事务数/概率；失效率增加=可用性降低 | 无：直接建模，预测最佳抗衰周期 | 用数学方法对历史数据建模，求解各状态驻留时间和转移概率；求最佳抗衰周期 | 考虑时间->考虑时间和负载;系统级停机、重启 |  | **1.优点：简便、易实施**  **2.不足：宏观、固定周期，不顾实际，造成不必要的恢复成本；**  **3.****依赖于失效先验知识，而实际上软件是在不断变化的（自身升级、更新，运行环境变化[17]）** |
| **基于度量/预测的方法（视情抗衰）** | | | | | | | | | |
|  | 1998 | Garg. S | A methodology for detection and estimation of software aging | 1. Unix工作站-操作系统 2. 剩余内存、已用交换空间、文件表大小、进程表大小、队列长度、磁盘数据块数目 3. 人工经验：线性、周期性依赖分析 | 直接预测 | 时序分析(趋势分析)-斜率评估技术：单指标预计耗尽时间； | 系统级重启、清理 | 预计耗尽时间 | 不确定性大，方法不具备普适性。对于软件衰退评估和性能预测来说，仅利用故障数据是不够的  忽略了负载的影响。 |
|  | 1999 | K Vaidyanathan, K.S. Trivedi | A measurement-based model for estimation of resource exhaustion in operational software systems |  | 基于度量的方法。 | 基于趋势分析的性能预测 |  | semi Markov |  |
|  | 2000 | T Dohi, K．Goseva-Popstojanova, K．S．Trivedi | Statistical non-parametric algorithms to estimate the optimal software rejuvenation schedule |  |  | 基于趋势分析的性能预测 |  |  |  |
|  | 2001 | V. Castelli, R.E. Harper, P. Heidelberger, K.S. Trivedi, K. Vaidyanathan | Proactive management of software aging |  |  |  |  |  |  |
|  | 2001 | Kalyanaraman Vaidyanathan,  , Richard E. Harper, Steven W. Hunter, Kishor S. Trivedi | Analysis and Implementation of Software Rejuvenation in Cluster Systems |  |  |  |  |  |  |
|  | 2002 | Vaidyanathan, K.  博士论文 | Proactive management of software systems: Analysis and implementation | 集群系统 |  | 预计耗尽时间。  基于度量的方法预测衰退相关的失效，以自动触发抗衰活动解决潜在的问题；基于分析的方法（如基于时间的方法）用于决定最优抗衰实施时间，通过开发和分析随机模型，最大化可用性或最小化停机代价。 | 用随机前向网为集群系统建模和分析不同的抗衰策略。  也为可以通过一些观测指标决定衰退级别的系统构建基于检查的预防性维护模型，模型用马尔科夫再生过程理论求解，得到最优抗衰策略 |  |  |
|  | 2002 | Li L., K. Vaidyanathan, K.S. Trivedi | An Approach for Estimation of Software Aging in a Web Server |  |  | 基于趋势分析的性能预测.时序分析-ARMA模型 |  |  | 假设软件衰退有一个总的趋势；基于衰退相关的参数的先验知识[] |
|  | 2002 | [Cassidy, K.J.](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=Authors:.QT.Cassidy,%20K.J..QT.&newsearch=partialPref);   [Gross, K.C.](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=Authors:.QT.Gross,%20K.C..QT.&newsearch=partialPref);   [Malekpour, A.](http://ieeexplore.ieee.org/search/searchresult.jsp?searchWithin=Authors:.QT.Malekpour,%20A..QT.&newsearch=partialPref); | Advanced pattern recognition for detection of complex software aging phenomena in online transaction processing servers | 1. OLTP servers 2. 共享内存池latch contention(抢占锁) 3. 训练模式识别，减少维数 | 商业化的设备状况监测系统（*SmartSignal eCMTM*）从长期监测的正常行为数据中构建经验模型；比对预测值和实际值的差别；训练模式识别算法，检测衰退 |  | 无 |  | 没特别读懂。之前以为它用模式识别来提取能刻画衰退的一般性特征；实际只有针对性地刻画反应共享内存池抢占锁导致衰老的问题；不足：没有考虑参数间的关联性，实验结果显示预测误报率有点高，需要额外方法确认、纠错 |
|  | 2003 | Y. Bao, X. Sun, K.S. Trivedi | Adaptive Software Rejuvenation: Degradation Model and Rejuvenation Scheme | 1. 系统资源泄露：内存泄露 2. 自适应衰退模型：可以用基于模型的、也可以用基于度量的方法 | 自适应评估模型： | 自适应评估模型： | 自适应抗衰安排策略 |  | 现有的大部分方法都缺乏固定的/一致的控制基础[] |
|  | 2003 |  | Proactive detection of software aging mechanisms in performance critical computers |  | 软件衰退机制的预防式检测：性能衰退前检测衰退的incipience and onset |  |  |  |  |
|  | 2003 | R. Sahoo, Oliner, A.; Rish, I.; Gupta, M.; Moreira, J.; Ma, S.; Vilalta, R. & Sivasubramaniam, A. | Critical event prediction for proactive management in large-scale computer clusters |  |  | 预测大规模聚类中的关键事件和系统参数：不同的时序分析方法和基于规则的分类算法 |  |  |  |
|  | **2003**  **[M]** | K.Y. Cai, and J.W. Cangussu, R.A. DeCarlo and A.P. Mathur | **An overview of software cybernetics（控制学）** |  | **原来研究者认为软件与硬件不同，不会老化，但****研究表明软件也会老化，本文提到软件老化不容忽视，尤其是网络环境下的软件，其老化会导致服务器挂起或崩溃；概述了潜在的导致软件老化的原因；作者提出****现有的大部分方法都缺乏固定的/一致的控制基础，作者提出疑问：软件抗衰是一个软件控制学问题？如何适应性地执行抗衰？** | **定期或在软件服务挂起或崩溃前实施抗衰；研究内容包括：实时监测软件当前状态、收集信息、决定何时和软件的哪部分应该抗衰。服务的性能和质量对网络软件来说很重要，网络软件也应该调整它的运行和服务情况，以与网络流量的变化一致。** |  | **Software rejuvenation adaptively** | **综述性论文，提出软件抗衰是潜在的研究方向，未提出确切的方法** |
|  | 2004 | Wei Xie,Yignang Hong, K.S. Trivadi | Software Rejuvenation Policies for Cluster Systems under Varying Workload． |  |  | 基于建模技术的性能预测： |  |  |  |
|  | 2004 | M. Malek E Salfner, G.A. Hoffmann | Self Rejuvenation：An Effective Way to High Availability |  |  | 基于建模技术的性能预测： |  |  |  |
|  | 2005 | Bao, Y.; Sun, X. & Trivedi, K. S. | A workload-based analysis of software aging and rejuvenatio | 资源泄漏 |  | 非齐次马尔柯夫链：构建衰退模型 | 考虑时间和工作负荷的；半马尔科夫过程 | elusive |  |
|  | 2005 | M. Grottke and K.S. Trivedi | Software Faults, Software Aging and Software Rejuvenation |  | 解释了（即使代码中没有额外的故障）衰退是如何发生的，以及若干种bug及其对衰退的影响 |  |  |  |  |
|  | 2005 | Vaidyanathan, K. & Trivedi, K. S. | A comprehensive model for software rejuvenation |  |  |  |  |  |  |
|  | 2006 | M.Grottke; L.Li; K.Vaidyanathan, K. S. Trivedi | Analysis of Software Aging in a Web Server | 1. Web server: apache. 2. 内存泄露（剩余物理内存和交换空间使用情况）和apache运行性能（responseTime）数据 3. 人工经验;考虑了单个资源趋势的周期性 |  | 用数个不同的非参数化统计方法，计算历史数据中的衰退趋势；然后用时序分析方法预测未来任何资源的值，并计算它是否会耗尽。在预测资源消耗时，考虑了数据的周期特性（关注点：如何建模周期特性、如何确定模型order） |  |  | 用已知的异常数据拟合[],不足以发现未知异常、衰退 |
|  | 2006 | R. Matias Jr. , Paulo J. F. Filho | An Experimental Study on Software Aging and Rejuvenation in Web Servers |  | 衰退对web server的影响 |  |  |  |  |
|  | 2006 | Seong (Steve) R. Yu | A software replication model for rejuvenation transparency to clients in a single computer environment | 单服务器环境的 |  |  | 热备抗衰技术=抗衰+完整冗余(hot-passive软件复制），使单服务器环境下，抗衰实施不用刻意避开高峰期 |  |  |
|  | 2006 | 徐建 | 软件抗衰若干关键技术研究(PhD thesis) |  |  |  |  |  |  |
|  | 2006 | Silva, L. and Madeira, H. and Silva, J.G. | [Software Aging and Rejuvenation in a SOAP-based Server](../../00-参考论文/软件衰退%20抗衰/(2006)%20Software%20Aging%20and%20Rejuvenation%20in%20a%20SOAP-based%20Server.pdf) |  |  |  |  |  |  |
|  | 2007 | 徐建 | 软件抗衰研究综述 |  |  | 预测软件性能：认为时序分析方法从本质上讲都是线性模型，且统计学方法大都需要一些难以证明的显眼知识。故一些较为复杂的时间序列(如软件的性能预测数据)，这类模型的预测结果不能满足应用的需求。 |  | Heisenbug(瞬时性、隐蔽性bug) | 常见的清理内部状态的方法：清空缓存、内存垃圾收集，重新初始化内核表、清理文件系统、收集碎片等 |
|  | 2007 | Luis Silva, A.Andrzejak | Deterministic Models of Software Aging and Optimal Rejuvenation Schedules | 1. SOAP server Apache Axis 1.3(中间件). 2. 请求率能力=function(服务的请求数). 3. 动态、自动为指定的致衰indicators（y=f(x) ,只考虑一个指标x）找指标，并用统计方法自动验证其正确性。比较通用、参数少 |  | 自适应抗衰？ |  | 为指定的致衰因素找最适合的一个指标。评价上述方法的方法可能值得借鉴 | 本文方法成立的假设：大多数情况，衰退过程能够被一个确定的衰退模型足够精确的捕获；且假设性能下降的主要indicators可以被逼近为一些“work”相关的度量标准（metric），如自前一次抗衰起服务的请求数。如果内存泄露或其他资源未释放是衰退的主要原因的话。而且只考虑了单个指标，没有考虑多指标的关联(复杂的衰退情况是有关联的)；且未考虑时间和软件负荷 |
|  | 2008 | A.Andrzejak, L. Silva | Using machine learning for non-intrusive modeling and prediction of software aging | 1. SOA应用-应用服务器 2. 处理的最大请求数/秒 3. 求每秒能处理的最大请求数 | 机器学习 | 机器学习方法(决策树、支持向量机、朴素贝叶斯)建模和预测确定性软件异常 |  | non-intrusive:外置，非内置； transient; adaptive, robust | 作者未验证其方法在动态设置或引起软件异常的多资源耗竭上的效果[] |
|  | 2008 | Cherkasova, L.; Ozonat, K.; Mi, N.; Symons, J. & Smirni, E. | Anomaly? Application Change? or Workload Change? Towards Automated Detection of Application Performance Anomaly and Change |  |  |  |  |  | 假设软件可用静态模型建模，模型只依赖工作负荷且不会衰退。 |
|  | 2010 | Javier Alonso and Jordi Torres, J osep Ll. Berral and Ricard Gavalda | Adaptive on-line software aging prediction based on Machine Learning |  |  |  |  |  |  |
|  | 2011 | Kourai, K. & Chiba, S. | Fast Software Rejuvenation of Virtual Machine Monitors | 虚拟机系统中的虚拟机监测器(VM monitor) |  |  | warm-VM reboot |  |  |
|  | 2011 | Alonso, J.; Belanche, L. & Avresky, D. R. | Predicting Software Anomalies Using Machine Learning Techniques | 1. 多层次在线书店(Apache Tomcat& mysql); 2. 软件失效 3. **Lasso 正规化技术（稀疏回归方法）,减少用于构建预测模型的监测指标的数目(最好情况:60%)** | 把预测转成检测状态异常;  机器学习算法，将系统状态聚成3类：绿（ok）、黄（警告，红色警告出现前5分钟）、红（危险，软件崩溃前5分钟） | 机器学习算法( R statistical language, Rpart（Decision Tree）,Naive Bayes,Support Vector machines Classifiers,K-nearest neighbors(knn),**Random Forest(随机森林)**, LDA（线性决策分析）/QDA（二次决策分析）),选择依据：拟合测试数据最好的 |  | 预测资源耗尽引起的异常并非易事；动态和不确定性软件异常；资源消耗曲线非线性；趋势动态变化； | 预测剩余寿命（time until crash），而非剩余可用寿命，因为没考虑可用性；  预测剩余寿命太难，转为检测即将发生的失效的告警；具体ML预测算法的选择依赖于训练、测试数据的选择，量又不大。特征提取的目的是为了降低模型的构建成本和复杂度，以及采集任务的难度，而不是考虑参数间的关联对分析结果的影响； |
|  | 2011 | Jean Araujo, Rubens Matos, Paulo Maciel,Rivalino Matias, Ibrahim Beicker | Experimental Evaluation of Software Aging Effects on the  Eucalyptus Cloud Computing Infrastructure | 1. Eucalyptus云计算平台 2. 内存泄露和内存碎片 | 线性回归 | 无 | 手动发送指令 |  | 仅用实验评估衰退对云计算平台的影响。未来准备通过可靠性度量来处理衰退的影响 |
|  | 2011  综述 | Cotroneo, Domenico and Natella, Roberto and Pietrantuono, Roberto and Russo, Stefano | Software Aging and Rejuvenation:Where We Are and Where We Are Going | 1. 安全关键的系统(7%)、非安全关键的系统(业务、任务:web server, DBMS) 2. 健康态、故障易发态、抗衰态；or**内存泄漏**(物理内存、交换空间)，**性能异常**(吞吐量、响应延迟)；or不具体某一种资源的耗竭。 3. **未来研究：**  * 分析更复杂的衰退表现/情况，其他类型的致衰bug：**算术错误(越界错误等)**、存储相关的bug、与管理system-depended数据结构相关的bug(如DBMS的共享内存池、JVM的虚拟内存等)； * 综合分析真实的致衰bugs. |  | 大多关注预测**失效时间**。分为:   1. 基于模型的分析方法：用马尔科夫决策过程和随机Petri网等方法,建立系统的数学模型,解最优抗衰实施时间 2. 基于度量的分析方法：用时序分析和机器学习识别资源耗竭和性能下降的趋势 3. **未来研究**  * 开发混合方法，实现在线监测和抗衰评估框架 | 1. **决策实施抗衰的最优安排：**  * 基于分析模型的方法(如基于时间的抗衰) * 基于度量的方法（如基于预测的抗衰）  1. **目前实施的抗衰技术：**  * 应用无关的（各级重启：组件、应用、VM、节点重启,集群） * 应用相关的：垃圾回收(如java的GC)、内核表刷新,需支持  1. **未来研究：**  * 更有效的抗衰技术，降低抗衰成本(在线、微重启) * 解释state of the application * 实现额外的机制，通过有选择的部分执行抗衰来最小化宕机时间 * OS和中间件、编程语言中融入抗衰策略 |  | 本文提出的未来应做的： |
|  |  |  |  |  |  |  |  |  |  |
|  | 总结 |  |  | 衰退的影响是逐步的性能下降和失效，不同的研究针对的影响不同，目前主要针对**内存泄漏**(物理内存、交换空间)和**性能异常**(服务率：吞吐量、响应延迟)。   1. **未来研究[34]：**  * 更多地在真实系统中实践 * 分析更复杂的衰退表现/情况，其他类型的致衰bug：**算术错误(越界错误等)**、存储相关的bug、与管理system-depended数据结构相关的bug(如DBMS的共享内存池、JVM的虚拟内存等)； * 综合分析真实的致衰bugs.   这就需要有一个通用的刻画软件可用性/状态的方法，能够智能的选择关键参数。 |  | 目前预测抗衰时机主要是针对失效(剩余寿命)，可以说抗衰是失效敏感的；而实际软件系统也比较关心性能。对于运行时软件失效的分布和性能下降的nature具有随机性。   1. **基于模型的分析方法**：用马尔科夫决策过程和随机Petri网等方法,建立系统的数学模型,解最优抗衰实施时间 2. **基于度量的分析方法**：用时序分析和机器学习识别资源耗竭和性能下降的趋势 3. **未来研究****[34]**  * 开发混合方法，实现在线监测和抗衰评估框架 * 预测抗衰时机不应只针对失效（热备、集群很普及，它们主要就是失效触发的），虽然这对抗衰不应只是失效触发的，也应该是性能耗竭触发的   这就需要预测时，不只是预测软件何时会失效（剩余寿命），还要根据软件性能要求(输入)，预测软件何时性能不够用(剩余可用寿命) | 负载等环境因素对软件的状态有较大影响。   1. **决策实施抗衰的最优安排：**  * 基于分析模型的方法(如基于时间的抗衰) * 基于度量的方法（如基于预测的抗衰）  1. **目前实施的抗衰技术：**  * 应用无关的（各级重启：组件、应用、VM、节点重启,集群） * 应用相关的：垃圾回收(如java的GC)、内核表刷新,需支持  1. **未来研究[34]**  * 更有效的抗衰技术，降低抗衰成本(在线、微重启) * 解释state of the application * 实现额外的机制，通过有选择的部分执行抗衰来最小化宕机时间   OS和中间件、编程语言中融入抗衰策略 |  |  |
|  | **抗衰实施技术** | | | | | | | | |
|  | 2004 | G. Candea ,S. Kawamoto, Y. Fujiki, G. Friedman,Fox, A. | Microreboot - A technique for cheap recovery  (ACM) |  |  |  |  |  |  |

软件抗衰相比软件冗余等事后容错方法，是一种预防性的容错方法；它也被称为“软件再生”。

1. 软件衰退发现与应对要解决的本质问题
2. 现有方法已解决的本质问题
3. 未来研究尚需要解决的关键问题
4. 软件老化潜在的原因

软件老化潜在的原因有内存泄露、未释放的文件锁、未终止的线程的累积、数据越界错误的增长、文件系统碎片、共享内存池封锁（死锁？）等等[18]

1. 徐建，软件抗衰若干关键技术研究，2007

数据分析方法：

1、朴素贝叶斯：可针对多行为建模？