基于统计学的软件系统预恢复时间阈值计算策略*)

王纪文 徐 建 游 静 刘凤玉
(南京理工大学计算机系 南京 225300)

摘 要 软件系统的预恢复是一种预防和主动的容错技术。本文提出了一种基于统计学的软件系统自恢复时间阈值计算算法。该算法基于满足软件系统可用性概率最大化的思想,在系统性能衰退时间分布未知的情况下,根据一定量的性能衰退的检测数据,计算出优化的软件系统自恢复时间阈值。仿真实验结果表明计算结果合理、稳定性好,能有效地应用于实际系统中。

关键词 软件系统预恢复,统计估计,性能衰退时间分布,系统可用性

A Statistical Strategy to Estimate the Optimal Software Rejuvenation Schedules

WANG Ji-Wen XU Jian YOU Jing LIU Feng-Yu
(Dept. of Computer, Nanjin University of Science and Technology, Nanjin 225300)

Abstract Software rejuvenation, a preventive maintenance technique for software fault tolerance, has been extensively studied in the recent literature. In this paper, we develop a statistical algorithm to estimate the optimal software rejuvenation schedules, provided that the statistical complete sample data of failure times is given, and the optimal software rejuvenation schedules which maximize the system availabilities are derived analytically. Numerical examples illustrate the resulting estimators for the optimal software rejuvenation schedules have quite nice convergence properties and are useful in applying to a real software system in operation without specifying the underlying failure time distributions.

Keywords Software rejuvenation, Statistical estimation, Failure time distribution, Availability

软件系统的预恢复是指系统在性能衰退过程中通过寻求最佳恢复时间阈值重启系统使系统可用性达到最大的一种性能恢复机制。对这方面的研究主要有两种方法:基于测量的恢复策略[1,2]和基于模型的恢复策略[3]。基于测量的恢复策略是通过对系统的检测积累的性能衰退数据进行分析得到最佳恢复时间阈值的技术;基于模型的恢复策略则是建立系统性能衰退的模型,根据性能衰退的分布函数得到最佳恢复时间阈值的技术。基于测量的恢复策略需要大量的检测数据的积累,而基于模型的恢复策略则需要得到性能衰退的分布函数,在大多数情况下其分布往往很难确定。本文提出了一种基于统计学的自恢复时间阈值计算策略,根据一定量的性能衰退数据计算出满足系统可用性最大的时间阈值。

1 模型描述

将软件系统的系统性能衰退和复原过程看作是连续时间的半马尔可夫过程,定义四个状态[3]:

0:系统正常运行状态;

1:系统可能发生故障的状态;

2:系统发生故障状态;

3:软件系统性能恢复状态。

系统的状态转换如图1所示。假设所有上述状态均为可更新状态;设随机变量X表示0→1的状态转换时间,其分布设为$P_r\{X\leqslant t\}=F_0(t)$,其均值为$u_0(>0)$;Y表示1→2的状态转换时间,分布为:$P_r\{Y\leqslant t\}=F_f(t)$,有限均值为:$\lambda_f(>0)$,Z表示系统从2→0状态的转换时间,且分布为$P_r\{Z\leqslant t\}=F_a(t)$均值为:$u_a(>0)$;1→3和3→0状态的转换分布分别为:$F_r(t)$和$F_c(t)$。

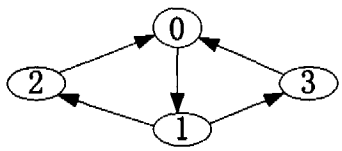

图1 系统性能衰退和恢复状态转换图

软件系统在性能衰退过程中,如果选择合适的时间阈值进行性能恢复就能够使系统的可使用时间达到较大或最大。

设t_0为系统性能恢复的时间阈值,则有:

$$F_r=U(t-t_0)=\begin{cases}1,\text{if } t\geqslant t_0\\0,\text{otherwise}\end{cases}$$

系统运行在可能产生故障之后,系统故障会以一定的概率发生,如果系统故障发生在予恢复发生之前,则故障恢复过程开始执行。否则,软件予恢复启动。软件从正常运行状态到下一个正常运行状态到来的时间为一个周期。

2 可用性概率计算

由模型分析可知:系统的可用性指的是系统处于可运行状态的概率,它包括系统处于0状态以及0→1状态转换的概率之和[3~5],故有:

$A(t_0)$=Pr{软件系统处于可运行状态的概率}

*)本课题得到国家自然科学基金(No. 60273035)资助。王纪文 博士研究生,主要研究领域为软件自愈与抗衰,信息安全。徐 建 博士研究生,主要研究领域为软件自愈与抗衰,信息安全。游 静 博士研究生,主要研究领域为软件自愈与抗衰,信息安全。刘凤玉 教授,博士生导师,主要研究领域为人工智能和网络安全。

$$=\lim_{t\to\infty}\{P_{00}(t)+P_{01}(t)\}$$

$$=\frac{u_0+\int_0^{t_0}\overline{F}_f(t)\mathrm{d}t}{u_0+u_aF_f(t_0)+u_c\overline{F}_f(t_0)\mathrm{d}t+\int_0^{t_0}\overline{F}_f(t)\mathrm{d}t}=S(t_0)$$

$$/T(t_0),$$

假定：$u_a>u_c$。

即软件发生故障之后的恢复时间大于软件在发生故障之前所进行的性能予恢复时间，对 $A(t_0)$ 进行分析：

(1)假设 $F_f(t)$ 为非降函数时，定义非线性函数：

$Q(t_0)=T(t_0)-\{(u_a-u_c)R_f(t_0)+1\}S(t_0)$，

其中，$R_f(t)=(\mathrm{d}F_f(t)/\mathrm{d}t)/\overline{F}_f(t)$ 是故障发生率。

(i)如果 $Q(0)>0$ 且 $Q(\infty)<0$，

则存在 t_0^* $(0<t_0^*<\infty)$，满足：$Q_1(t_0^*)=0$ 使得 $MaxA(t_0^*)=1/(u_a-u_c)R_f(t_0^*)+1$

(ii)如果 $Q(0)\leqslant 0$，则有：$t_0^*=0$，使得：$MaxA(0)=u_0/(u_0+u_c)$

(iii)如果 $Q(\infty)\geqslant 0$，则有：$t_0^*\to\infty$，使得：$MaxA(\infty)=(u_0+\lambda_f)/(u_0+u_a+\lambda_f)$

(2)假定 $F_f(t)$ 为非升函数时，

$A(t_0)$ 是 t_0 的凸函数，且优化的系统性能恢复为：$t_0^*=0$ 或 $t_0^*\to\infty$。

3 统计的算法实现

设 $F_r(t)$ 为软件系统性能衰退的分布函数，考虑到满足 $Max_{0\leqslant t\leqslant\infty}A(t^*)$ 的 t^* 的可解性，根据文[6～9]，可得衰退时间分布的等价转换形式为：

$$\phi(p)=(1/\lambda_f)\int_0^{F_f^{-1}}\overline{F}_f(t)\mathrm{d}t,$$

其中：$F_f^{-1}(p)=\inf\{t_0:F_f(t_0)\geqslant p\}$；$(0\leqslant p\leqslant 1)$。

令：$\alpha=\lambda_f+u_0$，$\beta=u_c*(u_a-u_c)$。

由于 $\phi(p)$ 在[0,1]区间内具有和 $F_r(t)$ 一致的凹凸性，考虑到性能衰退时间的期望值 λ_f 以及 u_0，u_a 和 u_c 的影响，则得变形的 t_0^* 的计算公式为：

满足 $\max\limits_{0\leqslant p\leqslant 1}\dfrac{\phi(p)+\alpha}{p+\beta}$ 的 $t_0^*=F_f^{-1}(p^*)$，t_0^* 为最佳预恢复时间阈值。

设 $x_1,x_2,\cdots,x_n$ 是检测到的系统性能衰退的一组数据，服从未知的连续分布 $F_r(t)$，且 $0=x_0\leqslant x_1\leqslant x_2\leqslant\cdots\leqslant x_n$，由检测到的性能衰退数据构造等价于 $\phi(p)$ 的计算公式如下，定义：$\phi_{nj}=\psi_j/\psi_n$，其中 $\psi_j=\sum_{k=1}^{j}(n-k+1)(x_k-x_{k-1})$，$(j=1,2,\cdots,n;\psi_0=0)$。而 $x_k,k=1,\cdots,j$ 是系统性能衰退的检测数据，将经验分布函数 $F_n(x)$ 和上述公式的计算结果联系起来，得到近似的分布：

$F_n(x)=\begin{cases}j/n, x_j\leqslant x\leqslant x_{j+1};\\ 1,\text{for}: x_n\leqslant x。\end{cases}$ 其中 $F_n(x)$ 是概率分布函数。

对比 $\phi(p)$ 和 ϕ_{nj}，则可得到可计算的最佳时间阈值为：

t_0^* 由 x_j^* 给出，其中：

$$j^*=\left\{j\,|\max_{0\leqslant j\leqslant n}\frac{\phi_{nj}+\alpha_j}{j/n+\beta}\right\};$$

λ_f 由公式 $\sum_{k=1}^{n}xk/n$ 代换。实验的数据证明见第 4 节。

4 仿真试验结果

本实验采用由 Weibull 分布随机产生的性能衰退数据，由第 3 节中的统计算法来计算最优化的预恢复时间阈值。图 2 中：取 $\theta=0.923$，$\beta=4.35$，$u_0=2.0$，$u_a=0.04$，$u_c=0.03$，计算可得其时间阈值为：$t^*=0.59572$，软件系统的可用性概率为：$A(t^*)=0.98470$。

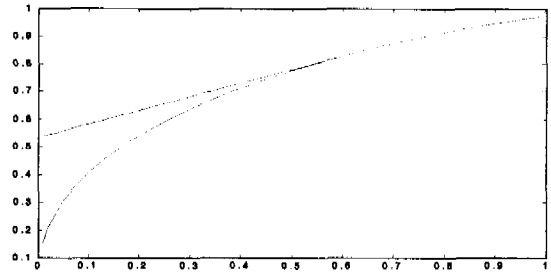

图 2 ϕ_{nj} 随 p 变化的关系图

表 1 是根据不同的 u_a/u_c 比值的不同计算所得的自恢复时间阈值，以及在此时间阈值下的软件系统可用性概率估计。其中：$u_0=240$，$u_c=0.3$，$\lambda_f=2180$，$\beta=2.35$

表 1 u_a/u_c 比值变化时 t^* 和 $A(t^*)$ 的变化关系

u_a/u_c	t^*	$A(t^*)$	u_a/u_c	t^*	$A(t^*)$
2	2416.9	0.999774846	7	841.6	0.999369824
3	1591.4	0.999684812	8	768.7	0.999324841
4	1260.3	0.999594795	9	710.3	0.999234889
5	1066.5	0.999504794	10	662.5	0.999144953
6	938.2	0.999459800	11	621.9	0.999099991

图 3 所示为基于统计学概率估计的可用性概率值 $A(t^*)$ 随性能衰退的数据个数 n 的变化表示。其中 $\theta=0.903$，$\beta=4.15$，$u_0=2.0$，$u_a=0.04$，$u_c=0.03$。

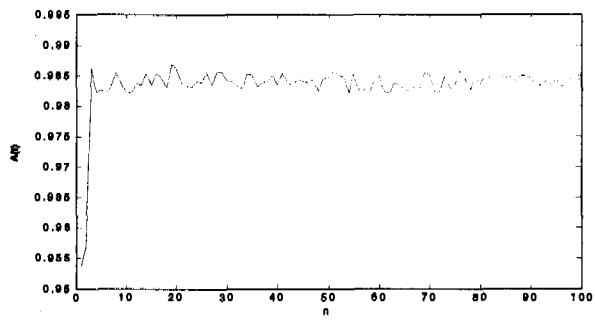

图 3 $A(t^*)$ 和 n 的函数变化关系

结束语 针对软件系统运行初期难以获取大量的性能衰退数据，而且性能衰退的分布也难以确定的情况，本篇论文提出了一种基于统计学的系统最佳预恢复时间阈值计算算法。仿真试验结果表明当性能衰退数据达到 $n=40$ 时，对最佳时间恢复阈值的计算估计值趋于稳定。该算法在软件系统运行时，根据少量的数据能够得到最佳时间恢复阈值的近似值，对于软件系统的维护，研究性能衰退的分布规律，构筑系统的自恢复模型，有实际意义。

参 考 文 献

1 Vaidyanthan K，Trivedi K S. A Measurement-Based Model for Estimation of Resource Exhaustion in Operational Software Systems. 10th International Symposium on Software Reliability Engineering, 1999. Boca Raton, Florida

(下转第 265 页)

软件开发方法学的角度提出了 MDA。MDA 将系统的功能规约和系统功能在特定平台上的实现规约相分离，从而将技术与平台变化对系统的影响降低到最小程度。基于 MDA 开发的系统可以方便地实现现有系统交互、集成和系统移植，也可以方便地集成未来新技术，以及最大程度地实现模型复用，降低软件开发成本和保护现有投资。

我们认为 Web Services 和 MDA 的结合将会给异构系统的交互和集成提供理想的解决方案，因此研究基于 MDA 的 Web Services 开发是非常有价值的。本文分析了目前软件开发面临的巨大挑战，指出了研究动因，从 Web Services 和 MDA 的基本概念到二者结合的研究做了全面的阐述和分析，并提出了面向 Web Services 的模型驱动开发框架 MDDF4WS。

在今后的研究工作中，我们会逐步实现 MDDF4WS 中的转换算法，并使之成为真正可以运行的系统。

参 考 文 献

1 Carman M, Serafini L, Traverso P. Web Service Composition as Planning. In: Proc. of the Workshop on Planning for Web Services. Trento, Italy, June 2003

2 Martin J, Arsanjani A, et al. Web Services: Promises and Compromises. Queue, 2003, 1(1): 48～58

3 Staab S, van der Aalst W, et al. Web Services: Been There, Done That? IEEE Intelligent Systems, 2003, 18(1): 72～85

4 Soley R, the OMG Staff Strategy Group. Model driven Architecture. Object Management Group White Paper. November 27, 2000

5 W3C. Web Services Activity. http://www.w3.org/2002/ws/. Sept. 2004

6 Birbeck M, et al. Professional XML. 2nd ed. Wrox Press Inc, 2001

7 HP. web_services_tech_overview. www.hpmiddleware.com/downloads/pdf/web_services_tech_overview.pdf. Sept. 2004

8 http://www.w3.org/TR/wsdl/. Sept. 2004

9 http://www.uddi.org/. Sept. 2004

10 http://www.w3.org/TR/SOAP/. Sept. 2004

11 Kreger H. Web Services Conceptual Architecture (WSCA 1.0). IBM Software Group. May 2001

12 Tsur S, Abiteboul S, et al. Are Web Services the next revolution in E-Commerce? In: Proc. of the 27th Int'l Conf on Very Large Data Bases. Roma: Morgan Kaufmann Publishers, 2001. 614～617

13 http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#whatis. Sept. 2004

14 Systinet. Web Services: A Practical Introduction. A Systinet White Paper. www.systinet.com, Sept. 2003

15 Middleware Company. Model Driven Development for J2EE Utilizing a Model Driven Architecture (MDA) Approach. Middleware Company Technical report. www.middleware-company.com/casestudy. June 2003

16 Klepper A, Warmer J, Bast W. MDA Explained : The Model Driven Architechure: Practice and Promise. Addision Wesley, ISBN 0-321-19442-X, April 21, 2003

17 Gerber A, Lawley M, et al. Transformation: The Missing Link of MDA. First International Conference on Graph Transformation (ICGT2002), Barcelona , Spain. October 2002

18 Will Provost. WSDL First . http://webservices.xml.com/pub/a/ws/2003/07/22/wsdlfirst.html. Sept. 2003

19 Will Provost. UML for Web Services. http://www.xml.com/pub/a/ws/2003/08/05/uml.html. Sept. 2003

20 Bezinvin J, Hammoudi S, et al. Applying MDA Approach for Web Services Platform. In: Proc of the 8th IEEE Int'l Enterprise Distributed Object Computing Conf (EDOC 2004). Monterey, California, USA . Sept2004. 20～24

21 Keith Mantell. From UML to BPEL. www.ibm.com/developerworks/ webservices/library/ws-uml2bpel/. Sept. 2004

22 OMG. EDOC Part II v1.0, Annex E - Technology mappings from EDOC to Distributed Component and Message Flow Platform Specific Models. OMG Document Number: ad/2001-08-20

23 OMG. UML Profile for Enterprise Distributed Object Computing Specification(EDOC). OMG Document Number: ptc/2001-12-04

24 Patrascoiu O. Mapping EDOC to Web Services using YATL. In: Proc. of the 8th IEEE Int'l Enterprise Distributed Object Computing Conf (EDOC 2004). Monerey, California, USA, Sept. 2004

25 Kath O, Blazarenas A, et al. Towards Executable Models: Transforming EDOC Behavior Models to CORBA and BPEL. Proc of the 8th IEEE Int'l Enterprise Distributed Object Computing Conf(EDOC 2004). Monterey, California, USA, Sept 2004

26 Skogan D, Gronmo R, Solheim I. Web Service Composition in UML. In: Proc. of the 8th IEEE Int'l Enterprise Distributed Object Computing Conf. (EDOC 2004). Monterey, California, USA, Sept. 2004

27 Orriens B, Yang Jian, et al. Model Driven Service Composition. www.unitn.it/convegni/download/ icsoc03/papers/Bart Orriens.pdf. Sept. 2004

28 Business Rules Group. Defining business rules, what are they really?. http://www.brcommunity.com, July 2000

29 von Halle B. Business rules applied: Building Better Systems Using the Business Rule Approach. Wiley & Sons, 2002

30 Veryard R. Rule Based Development. CBDi Journal, July/August, 2002

31 de Castro V, Marcos E, Vela B. Representing WSDL with Extended UML. www.unab.edu.co/editorialunab/ revistas/rcc/pdfs/r51-art1 c.pdf. Sept. 2004

32 Kath O, Soden M, et al. An Open Modeling Infrastructure integrating EDOC and CCM. In: Proc. of the 8th IEEE Int'l Enterprise Distributed Object Computing Conf (EDOC 2003). Brisbane, Australia, Sept. 2003. 198～207

(上接第 259 页)

2 A Methodology for Detection and Estimation of Software Aging. Sachin Garg, Aad van Moorsel, Kalyanaraman Vaidyanthan and Kishor S. Trivedi. In: Proceedings of The Ninth International Symposium on Software Reliability Engineering, 1998

3 Bao Yujuan, Sun Xiaobai, Trivedi K S. Adaptive Software Rejuvenation: Degradation Model and Rejuvenation Scheme. 2003 International Conference on Dependable Systems and Networks, 2003. San Francisco, California

4 Barlow R E, Campo R. total Time on test Processes and Applications to Failure Data analysis . reliability and Fault Tree Analysis (R. E. Barlow, J. fussell and N. Singpurwalla, eds.), SIAM, Philadelphia, 1975. 451～481

5 Okamura H, Fujimoto A, Dohi T, Osaki S, Trivedi K S. The Optimal Preventive Maintenance Policy for a Software System with Multi Server Station. In: Proc. 6th ISSAT Int'l Conf. Reliability and Quality in Design, 2000. 451～481

6 huang Y, Kintala C, KoletisN, Funton N D. Software Rejuvenation: Analysis, Module and Applications . In: Proc. 25th IEEE Int'l symp. On Fault Tolerant Computing, IEEE Computer Society Press , Los Alamitos, CA, 1995. 381～390

7 Candea G, Delgado M, Chen M, Fox A. Automatic failure-path inference: A generic introspection technique for software systems. In : Proc. 3rd IEEE Workshop on Internet Applications, San Jose, CA, 2003

8 Kephart J O, Chess D M. The vision of autonomic computing. Computer, 2003, 36(1): 41～52

9 Konstantinou A V, Yemini Y. Programming Systems for Autonomy. In: Proceedings of the Autonomic Computing Workshop fifth Annual International Workshop on Active Middleware Services (Ams'03)