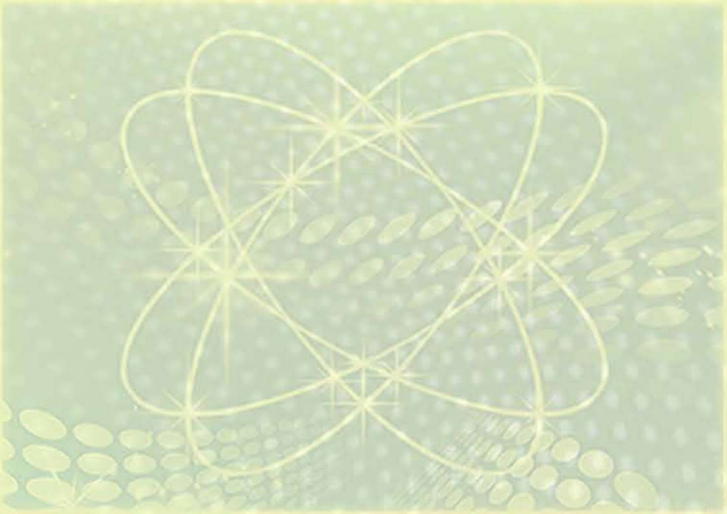
软件可靠性的不确定性研究

罗自强　曹鹏 著



湖北科学技术出版社

软件可靠性的不确定性研究

罗自强　曹鹏　著

湖北科学技术出版社

第一作者简介

罗自强，1977 年 12 月生于湖北咸宁，男，工学博士，中共党员。2007 年 9 月毕业于解放军理工大学，获工学博士学位。2009 年 9 月至今，在海南师范大学信息科学技术学院从事教学和科研工作。

研究方向：软件可靠性、复杂网络、人工智能等。

主持项目：海南省自然科学基金项目：基于云模型理论的软件可靠性评估方法研究，NO.611129 ；海南省高等学校科学研究项目（自然科学类重点项目）：基于复杂网络的数据挖掘方法研究，NO.Hjkj2012-14；海南师范大学博士科研启动资助项目：软件可靠性的不确定研究，NO.00203020215；海南师范大学 2011-2012 学年度实验室开放项目：基于 Markov 链的软件测试充分性的统计分析和实验模拟，NO.kfsy11047。

本书受海南省自然科学基金项目（NO.611129，NO.610223）、海南省高等学校科学研究项目（NO.Hjkj2012-14）、海南师范大学博士科研启动项目（NO.00203020215）、海南师范大学 2011-2012 学年度实验室开放项目（NO.kfsy11047）以及海南师范大学学术著作出版项目资助（NO.ZZ1122）。

前　言

在软件可靠性工程中，往往会遇到大量的不确定信息。本书首次利用云模型来统一刻画软件可靠性中蕴涵的随机性、模糊性及其关联性，构成软件可靠性的定性概念描述和定量数值表示间的映射，在此理论基础上提出了软件可靠性的定性评估新方法。

首先，基于概率统计理论对云作了进一步较深入的分析，从多个角度形象直观地展示了正态云；构造了一种新的精确度更高的逆向云算法；利用云表示软件可靠性的不确定性指标，建立了三种不同抽象层次的软件可靠性评价概念集；从定量的角度，验证了云模型对软件可靠性水平可以实现类似自然语言的软划分；这样，当得到某软件可靠性指标的虚拟云表示，就可通过 X 条件云发生器灵活地转换为软件可靠性的定性评价；阐述了传统软件可靠性模型及其局限性，进而提出用云对传统软件可靠性模型进行改进的构想。

其次，给出了 J-M 模型和两种非线性回归模型中参数及可靠性指标的云表示方式，分别结合经典的软件可靠性工程数据对软件作出了定性评价，并对比分析了传统方法和新方法。

再次，利用云表示软件系统的马尔可夫模型中无法确定的转移概率和模块的可靠性，建立了系统的云可靠度模型，并通过实例与传统方法进行了比较分析；基于云的代数运算分别给出了 AND 配置和 OR 配置软件系统的云可靠度公式，并通过实例与模糊方法进行了比较分析。

最后,综合考虑整个软件开发过程中影响软件可靠性的多种不确定因素,先简单修改了基于云模型的德尔菲层次法以应用于确定软件可靠性影响因素的权重,再使用标准加权关联规则挖掘法获得与软件可靠性相关的定性规则,从而基于云模型的不确定性推理可以实现软件可靠性的定性评价。

全书共8章,第1章为绪论;第2章为云模型与云进化策略;第3章为基于云模型的软件可靠性;第4章为基于云模型的软件可靠性预计模型的定性评估;第5章为基于MARKOV测试方法的软件可靠性评估;第6章为基于云模型的结构化软件的可靠性的定性评估;第7章为考虑多种不确定因素的软件可靠性的定性评估;第8章为结束语。另外,附录中详细介绍了有限状态、离散参数和时齐的MARKOV链。本书由罗自强(海南师范大学)和曹鹏(琼台师范高等专科学校)合著完成,其中第2章,第4章和第5章由罗自强执笔(10万字),第1章,第3章,第6章至参考文献由曹鹏执笔(8万字)。

光阴荏苒,转眼又是一春,有些欣慰,也有点遗憾;有些欢欣,也有点苦闷;……逝者如斯,但我始终所须坦然持有的是,希望和等待。

在本书完成之际,首先作者要向导师李德毅院士,张志华教授致以深深的敬意、由衷的感激之情和最诚挚的谢意。

本书的出版得到海南省自然科学基金(NO. 611129, NO.610223)、海南省高等学校科学研究项目(NO.Hjkj2012-14)、海南师范大学博士科研项目(NO. 00203020215)以及海南师范大学学术著作出版项目(NO.ZZ1122)的资助,在此深表谢意。

限于作者水平和时间仓促,书中难免存在不妥之处,敬请读者体谅和批评指正。同时欢迎读者信息交流。作者联系方式:571158 海口市龙昆南路99号 海南师范大学信息学院, E-mail: luo_letian@163.com。

作　者

2012年3月9日于海口

目　　录

第1章 绪 论

1.1 研究背景和意义

1.1.1 软件质量的重要性

21世纪，人类步入了信息时代，从交通、能源、电信到金融、教育、军事……等等大多数行业都需要计算机的辅助。软件是计算机系统的灵魂，是许多复杂系统的神经中枢，而质量则是软件的命脉。

软件失效造成系统瘫痪、人员伤亡以及重大经济损失的例子时有所闻。特别是一些安全关键软件，一旦发生失效就可能危及人的生命、财产和生态环境。

欧洲航天局 Ariane 5 号火箭发射失败是因为 Ada 语言在编译过程的检查失败导致的。将大的浮点数转换成整数是一种常见的程序错误来源。1996 年 6 月 4 日，对于 Ariane 5 火箭的初次航行来说，这样一个错误产生了灾难性的后果。发射后仅仅 37 秒，火箭偏离它的飞行路径，解体并爆炸了。火箭上载有价值 5 亿美元的通信卫星，还使耗资达 80 亿美元的开发计划推迟近 3 年。后来的调查显示，控制惯性导航系统的计算机向控制引擎喷嘴的计算机发送了一个无效数据。这件事可以说是历史上损失最惨重的

软件故障事件。

20世纪末,“千年虫”问题震惊世界,各国投入大量的人力和物力,耗资数千亿美元,虫灾才基本上得到控制。千年虫缩写为“Y2K”,又叫做“计算机2000年问题”,“2000年病毒”或“千年病毒”,是指在某些使用了计算机程序的智能系统(包括计算机系统、自动控制芯片等)中,由于其中的年份只使用两位十进制数来表示,因此当系统进行(或涉及到)跨世纪的日期处理运算时(如多个日期之间的计算或比较等),就会出现错误的结果,进而引发各种各样的系统功能紊乱甚至崩溃。

2003年8月14日,美国及加拿大部分地区发生了的史上最大停电事故,这是由位于美国俄亥俄州的第一能源(FirstEnergy)公司下属的电力监测与控制管理系统“XA/21”出现软件错误导致的。……这样的例子不胜枚举,诸如此类由于软件问题引发的事故基本上是由于软件自身质量造成的。

1979年,Fisher和Light将软件质量定义为:表征计算机系统卓越程度的所有属性的集合。1982年,Fisher和Baker将软件质量定义为:软件产品满足明确需求一组属性的集合。20世纪90年代,Norman、Robin等将软件质量定义为:表征软件产品满足明确的和隐含的需求的能力的特性或特征的集合。

根据计算机软件质量保证计划规范GB/T12504—1990中的定义,软件质量是指软件产品中能满足给定需求的各种特性的总和。这些特性称做质量特性,它包括功能性、可靠性、易使用性、时间经济性、资源经济性、可维护性和可移植性等。

ISO9126质量度量模型将质量特性划分为6个方面:

(1)功能性:适合性、准确性、互操作性、依从性和安全性;

(2)可靠性:成熟性、容错性和易恢复性;

(3)易使用性:易理解性、易学习性和易操作性;

(4)效率:时间特性和资源特性;

(5)可维护性:易分析性、易更改性、稳定性和易测试性;

(6)可移植性:适应性、易安装性、一致性和易替换性。

早在1976年,由Boehm等提出软件质量模型的分层方案。1979年McCall等人改进Boehm质量模型又提出了一种软件质量模型。McCall等认为,特性是软件质量的反映,软件属性可用做评价准则,定量化地度量软件属性可知软件质量的优劣。McCall[1]认为软件质量应由11个要素构成,即正确性、可靠性、效率、完整性、可使用性、可维护性、可测试性、灵活性、可移植性、可复用性和互连性,具体介绍 参见表1.1。又可分为面向产品运行、面向产品修改、面向产品转移三种类型,其相互关系如图1.1所示。

表1.1 McCall等人的软件质量特性定义

正确性	在预定环境下,软件满足设计规格说明书及用户预期目标的程度。它要求软件本身没有错误
可靠性	软件按照设计要求,在规定时间和条件下不出故障,持续运行的程度
效率	为了完成预定功能,软件系统所需的计算机资源的多少
完整性	为某一目的而保护数据,避免它受到偶然的或有意的破坏、改动或遗失的能力
可使用性	对于一个软件系统,用户学习、使用软件及为程序准备输入和解释输出所需工作量的大小
可维护性	为满足用户新的要求,或当环境发生了变化,或运行中发现了新的错误时,对一个已投入运行的软件进行相应诊断和修改所需工作量的大小
可测试性	测试软件以确保其能够执行预定功能所需工作量的大小
灵活性	修改或改进一个已投入运行的软件所需工作量的大小

续表

可移植性	将一个软件系统从一个计算机系统或环境移植到另一个计算机系统或环境中运行时所需工作量的大小
可复用性	一个软件(或软件的部件)能再次用于其他应用(该应用的功能与此软件或软件部件的所完成的功能有关)的程度
互连性	又称相互操作性。连接一个软件和其他系统所需工作量的大小。如果这个软件要联网或与其他系统通信或要把其他系统纳入到自己的控制之下,必须有系统间的接口,使之可以联结

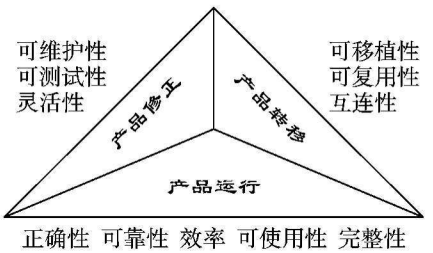


图 1.1 质量要素与产品状态

鉴于软件可靠性是软件质量最重要的固有特性之一,因此对软件可靠性评估进行广泛深入的研究,对提高软件质量以及合理控制软件开发过程有着重要的意义。

1.1.2 软件可靠性的不确定性

对于可靠性的研究最早可以追溯到 20 世纪 20、30 年代关于机器维修的问题,50 年代开始对于可靠性的研究进入黄金时代,但是这些研究仍然都是基于硬件的。

直到 70 年代由于软件危机的存在使得人们开始重视和研究软件可靠性问题。以 Jelinski,Shooman,Musa 等为代表的一批著名软件工程专家,援引可靠性工程学的观点,对软件可靠性作出定

义。他们的措辞虽不尽相同，但基调是一致的。另一批软件工程专家认为软件具有与硬件不同的性质，因而持反对态度。在这场争论中，JeIinski 等人的见解得到了系统工程师们的赞同，也逐渐得到了软件工程界多数人的支持。而且从目前的情况来看，这个定义在目前所有的定义中最恰当地反映了软件的可靠性特性，并对解决实际问题发挥了作用。关于软件可靠性的确切含义，学术界曾经有过长期的争论。1983 年美国 IEEE(Institute for Electrical and Electronic Engineers)计算机学会对“软件可靠性”一词正式给出了具有定量和定性双重含义的定义[1]：

(1)在规定的条件下，在规定的时间内，软件不引起系统失效的概率，该概率是系统输入和系统使用的函数，也是软件中存在错误的函数；系统输入将确定是否会遇到已存在的错误(如果错误存在的话)。

(2)在规定的时间周期内，在所述条件下，程序执行所要求的功能的能力。

在这个定义中，第一部分对软件可靠性进行了定量的描述，第二部分则对软件可靠性进行了定性的描述。因而软件可靠性具有定性的和定量的两层含义。在强调其定量的含义时，工程上常用软件的可靠度函数来说明软件的可靠性。可以看出，在这个定义中，软件和程序两个词汇被明显地混用了。这这个定义经美国标准化研究所批准作为美国的国家标准。1989 年，我国国标 GB /T—11157 采用了这个定义。

在该定义中，“规定条件”是指计算机的软、硬件环境，软件环境包括运行的操作系统、应用程序、编译系统、数据库系统等；硬件环境包括计算机的 CPU、CACHE、MEMORY、I/O 等。“规定时间”指软件的工作周期，有三种时间度量：日历时间(日、周、月、年)、时钟时间(程序运行从开始到结束时间的秒、分、时，它包括等待时间和其他辅助时间，但不包括计算机停机占用时间)和执行时

间(指计算机执行程序实际占用的中央处理器时间,故又称 CPU 时间)。“执行所要求的功能”通常指软件不出现失效。如果一个系统不能完成其功能,就说明它发生了失效。为了识别一个失效,必须明确要求它完成的功能是什么。规定的功能通常在软件需求规格说明书中定义。

传统的软件可靠性理论基于以下两个基本假设[3]:

(1)概率假设:系统可靠性行为可以完全用概率方式予以刻画。

(2)双态假设:系统只具有两个极端状态,完全正常状态和完全失效状态,系统在任意时刻必处于上述两种状态之一。

下面,我们将主要论述软件可靠性的两个基本的不确定性:随机性和模糊性。

1.1.2.1 软件可靠性的随机性

随机性是指有明确定义但不一定出现的事件中所包含的不确定性,它是由事物的因果关系不确定所造成的。概率论和数理统计就是研究和揭示随机现象的统计规律性的一门学科,它至今已有几百年的研究历史。

具有随机性的事件有以下一些特点:

(1)事件可以在基本相同的条件下重复进行,比如掷骰子。只有单一的偶然过程而无法判定它的可重复性则不称为随机事件。

(2)在基本相同条件下某事件可能以多种方式表现出来,事先不能确定它以何种特定方式发生,如不论怎样控制炮的射击条件,在射击前都不能毫无误差地预测弹着点的位置。只有唯一可能性的过程不是随机事件。

(3)事先可以预见该事件以各种方式出现的所有可能性,预见它以某种特定方式出现的概率,即在重复过程中出现的频率,如大量射击时炮弹的弹着点呈正态分布,每个弹着点在一定范围内有确定的概率。在重复发生时没有确定概率的现象不是同一过程的

随机事件。

宏观世界中必然发生的、确定性的事件在其细节上会带有随机性的偏离。微观世界中个别客体的运动状态都是随机的。对于一个随机事件可以探讨其可能出现的概率，反映该事件发生的可能性的大小。大量重复出现的随机事件则表现出统计的规律性。统计规律是大量随机现象的整体性规律，它支配着随机性系统的状态。

造成软件可靠性的随机性，有如下几个方面的原因：

(1)软件操作剖面(Operational Profile)的随机性

由于软件操作剖面的不确定性，即使一个软件通过测试并最终投放市场，它的可靠性也可能会随着环境因素的变化而发生变化。软件可靠性的定义表明，软件可靠性是面向用户的而不是面向开发人员的。软件可靠性与用户操作有关，而不是与程序的设计有关。操作剖面是指软件测试数据输入域，以及各种输入数据的组合使用概率。欧空局(ESA)标准 PSS－ 01－ 21(1991) “ESA 空间系统软件产品保证要求”，定义操作剖面为：“对系统使用条件的定义。系统的输入值都用其按时间的分布或按它们在可能输入范围内的出现概率的分布来定义”[4]。

假设软件的总输入域划分为若干个输入子域，那么这若干个子域都分别对应一个操作剖面。输入数据落入相应子域的可能性称为操作剖面值，显然操作剖面值会因为软件使用环境不同而发生变化。因为软件使用环境的不同，使得软件在不同环境下输入的数据也有所不同，造成了在不同环境下具有不同的剖面值，这样就形成了软件在不同环境下产生不同可靠性的现象。一般来说，操作剖面与发生概率之间的关系是软件输入空间越大，相应的操作剖面值则会越小[2]。

操作剖面在可靠性评估过程中有着重要的作用，准确的操作剖面能使我们得到准确、可信度高的评估结果，否则，结果的可信