火龙果•整理
uml.org.cn

软件可靠性度量

火龙果•整理
uml.org.cn

提纲

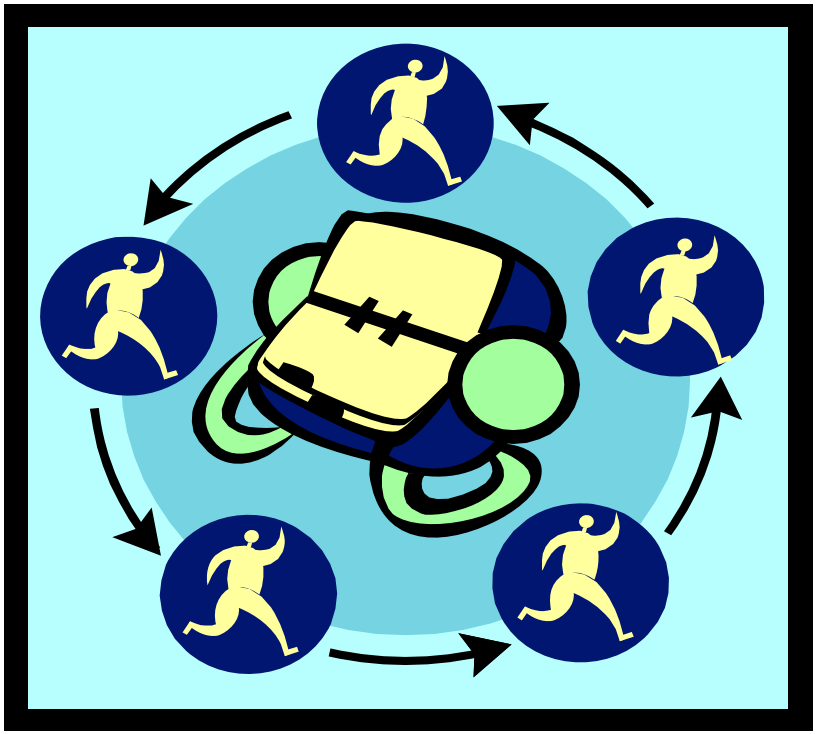
一. 软件可靠性度量概述

二. 软件可靠性模型的概念

三. 软件可靠性预计模型

方法

特殊——一般



火龙果•整理
uml.org.cn

三.软件可靠性预计模型(提纲)

1. JELINSKI-MORANDA模型
2. 软件可靠性预计模型的理论简介
3. 软件可靠性增长预计建模概况

1. JELINSKI-MORANDA模型

1.1 应用案例
1.2 基本假设
1.3 基本公式
1.4 参数的最大似然估计
1.5 参数的最小二乘估计
1.6 可靠性参数预计
1.7 模型的极限条件
1.8 FC型J-M模型
1.9 模型小结

1.1 应用案例

- 1972年，Jelinski和Moranda分析U.S. Navy Fleet Computer Programming Center的数据。这些数据是从美国海军战术数据系统 (Naval Tactical Data System-NTDS)保存的一个实时多处理机系统软件开发过程的资料中提取的，NTDS 的这一组数据包括了38个不同的模块，每个程序模块一般都经过了开发阶段、测试阶段和使用阶段。
- 每当操作者或用户发现了系统的错误征兆，就用软件异常报告的形式记载和反映，再由熟悉整个程序的人员跟踪这些问题，查找原因。 这些模块中的一个大型模块的失效间隔时间 (用日表示) 示于下表。
- 这个模块在开发阶段，发现了26个错误，在测试阶段又发现了5个错误，在使用阶段发现了1个错误后，对模块再次进行了测试，结果又发现了2个错误。

火龙果•整理
uml.org.cn

失效数据（1）

错误数（k）	实际失效时间（日）
1	9
2	12
3	11
4	4
5	7
6	2
7	5

火龙果•整理
uml.org.cn

失效数据（2）

错误数（k）	实际失效时间（日）
8	8
9	5
10	7
11	1
12	6
13	1
14	9

失效数据（3）

错误数（k）	实际失效时间（日）
15	4
16	1
17	3
18	3
19	6
20	1
21	11

失效数据（4）

错误数（k）	实际失效时间（日）
22	33
23	7
24	91
25	2
26	1
27	87
28	47

失效数据（5）

错误数（k）	实际失效时间（日）
29	12
30	9
31	135
32	258
33	16
34	35

- Jelinski和Moranda利用表中的前26个数据来拟合模型，得到了参数估计值。

火龙果•整理
uml.org.cn

失效数据（1）

错误数（k）	实际失效时间（日）
1	9
2	12
3	11
4	4
5	7
6	2
7	5

如果n表示失效次数，那么 $n=?$

如果 x_i 表示第i次失效与第i－1次失效之间的时间间隔，那么 x_i 是否已知？

如果 $t=\sum_{i=1}^{n} x_i$ ，那么 t 是否可以求得？

参数的求解

$$\sum_{i=1}^{n}\frac{1}{N_0-i+1}=\frac{n}{N_0-\frac{1}{t}\sum_{i=1}^{n}(i-1)x_i}$$

$$\Phi=\frac{n}{N_0 t-\sum_{i=1}^{n}(i-1)x_i}$$

其中 $t=\sum_{i=1}^{n}x_i$

利用数值计算法可以得到 N_0 和 Φ 的点估计值。

基本公式

失效率函数为

$$Z(x_i) = \Phi(N_0 - i + 1)$$

其分布函数为

$$F(x_i) = 1 - \exp\{-\Phi(N_0 - i + 1)x_i\}$$

其可靠性函数为

$$R(x_i) = \exp\{-\Phi(N_0 - i + 1)x_i\}$$

失效间隔时间为

$$MTBF_i = \frac{1}{\Phi(N_0 - i + 1)}$$

估计值

Jelinski和Moranda得到的参数估计值为 $\hat{N}_0 = 31.2$，$\hat{\Phi} = 0.00685$，失效间隔时间的估计值则可由下式算出：

$$MTB\hat{F}_i = \frac{1}{0.00685(32 - i + 1)}$$

火龙果•整理
uml.org.cn

失效数据（1）

错误数（k）	实际失效时间（日）	预测失效时间（日）
1	9	4.7
2	12	4.8
3	11	5.0
4	4	5.2
5	7	5.4
6	2	5.6
7	5	5.8

火龙果•整理
uml.org.cn

失效数据（2）

错误数（k）	实际失效时间（日）	预测失效时间（日）
8	8	6.0
9	5	6.3
10	7	6.6
11	1	6.9
12	6	7.2
13	1	7.6
14	9	8.0

火龙果•整理
uml.org.cn

失效数据（3）

错误数（k）	实际失效时间（日）	预测失效时间（日）
15	4	8.5
16	1	9.0
17	3	9.6
18	3	10.3
19	6	11.1
20	1	12.0
21	11	13.0

火龙果•整理
uml.org.cn

失效数据（4）

错误数（k）	实际失效时间（日）	预测失效时间（日）
22	33	14.3
23	7	15.9
24	91	17.8
25	2	20.3
26	1	23.5
27	87	28.1
28	47	34.8

火龙果•整理
uml.org.cn

失效数据（5）

错误数（k）	实际失效时间（日）	预测失效时间（日）
29	12	45.6
30	9	66.4
31	135	121.7
32	258	
33	16	
34	35	
总的错误数（34）		预计值＝32
失效率比例系数		=0.00685
残余错误数（8）		预计值＝32-26=5

火龙果•整理
uml.org.cn

应用案例结果分析

- 程序在使用和再测试中发现的错误数共有34个，假定程序的排错工作已经很彻底，可认为程序的固有错误总数为34个，估计值32略低于实际值。

1.2 基本假设

(1)程序中的固有错误数N_0是一个未知的常数。

(2)程序中的各个错误是相互独立的，每个错误导致系统发生失效的可能性大致相同，各次失效间隔时间 (即错误发生间隔时

有限故障类模型

(3)测试中检测到的错误，即 错只排除一个错误，排错时间可以忽略不计，在排错过程中不引入新的错误。

具有确定的错误数和故障数

1.2 基本假设（续）

(4)程序的失效率在每个失效间隔时间内是常数，其数值正比于程序中残留的错误数，在第i个测试区间，其失效率函数为

$$Z(x_i) = \Phi(N_0 - i + 1)$$

式中：Φ 为比例常数;

x_i 为第i次失效间隔中以i-1次失效为起点的时间变量，是一个随机变数。

(5)程序测试环境与预期的使用环境相同。

软件可靠性度量参数

- 可靠度 $R(t)=P(\xi>t)$
- 不可靠度 $F(t)=1-R(t)$
- 失效密度函数 $f(t)=dF(t)/dt$

软件可靠性度量参数

- 失效率：指在t时刻尚未发生失效的条件下，在t时刻后单位时间内发生失效的概率。

$$Z(t)=\lim_{\Delta t\to 0}\frac{P(t<\xi<t+\Delta t\,|\,\xi>t)}{\Delta t}=\frac{1}{R(t)}\cdot f(t)$$

- 失效强度：假设软件在t时刻发生的失效数为N(t)， 显然N(t)是一个随机数，且随时间t的变化而不同，即{N(t), t>0}为一随机过程。设u(t) 为随机变量N(t)的均值，则

$$\lambda(t)=\frac{du(t)}{dt}$$

软件可靠性度量参数

- MTTF:平均失效前时间

$$MTTF = \int_0^{\infty} R(t)\,dt$$

1.3 基本公式

- 运用可靠性工程学的基础理论，Jelinski和Moranda建立了J-M模型。

$$Z(t) = \Phi(N_0 - i + 1)$$

$$Z(t) = \frac{f(t)}{R(t)} = \frac{F'(t)}{R(t)} = \frac{-R'(t)}{R(t)}$$

$$\int_0^t Z(x)dx = \int_0^t \frac{-R'(x)}{R(x)}dx$$

$$\int_0^t Z(x)dx = \int_0^t -d[\ln R(x)] = -\ln R(x)\Big|_0^t = -\ln R(t)$$

$$R(t) = \exp\left[-\int_0^t Z(x)dx\right]$$

1.2 基本假设（续）

(4)程序的失效率在每个失效间隔时间内是常数，其数值正比于程序中残留的错误数，在第i个测试区间，其失效率函数为

$$Z(x_i) = \Phi(N_0 - i + 1)$$

式中：Φ 为比例常数;

x_i 为第i次失效间隔中以i-1次失效为起点的时间变量，是一个随机变数。

1.3 基本公式（续）

- 则可靠性函数为

$$R(t)=\exp[-\int_0^t Z(x)dx]=\exp[-\Phi(N_0-i+1)t]$$

- 则失效分布函数为

$$F(t)=1-\exp[-\Phi(N_0-i+1)t]$$

- 则失效密度函数为

$$f(t)=\Phi(N_0-i+1)\exp[-\Phi(N_0-i+1)t]$$

1.3 基本公式（续）

- 则平均失效前时间为

$$MTTF = \int_0^{\infty} R(t)dt = \int_0^{\infty} \exp[-\Phi(N_0 - i + 1)t]dt$$

$$= -\frac{1}{\Phi(N_0 - i + 1)} \exp[-\Phi(N_0 - i + 1)t]\Big|_0^{\infty}$$

$$= \frac{1}{\Phi(N_0 - i + 1)}$$

其中的参数如何获得？

1.3 基本公式（续）

- 则时间t之前的平均故障数为

$$\mu(t) = N_0 F_a(t)$$

式中：$F_a(t)$ 表示相应于单位错误失效率 $z_a(t)$ 的累积分布函数，对上式求导可以得到失效强度函数

$$\lambda(t) = N_0 f_a(t)$$

式中：$f_a(t)$ 是单位错误概率密度函数

失效强度函数

$$Z(t_i) = \Phi(N_0 - i + 1)$$

$$Z(t_{i+1}) = \Phi(N_0 - i)$$

$$Z_a(t) = \Phi$$

$$R(t) = \exp[-\int_0^t Z(x)dx]$$

$$R_a(t) = \exp[-\int_0^t Z_a(x)dx] = \exp[-\int_0^t \Phi dx] = \exp(-\Phi t)$$

$$F_a(t) = 1 - R_a(t) = 1 - \exp(-\Phi t)$$

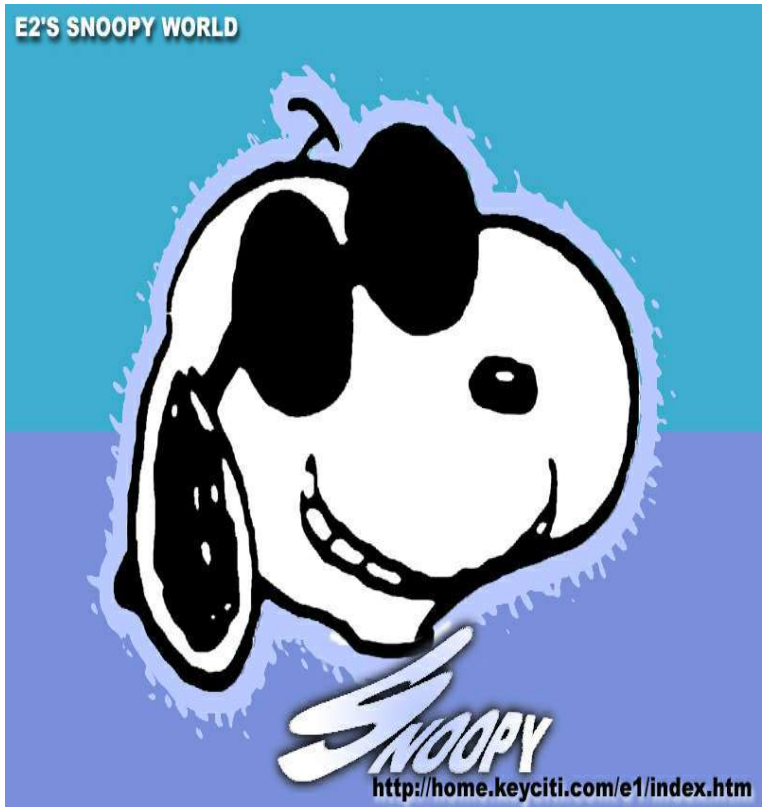
$$f_a(t) = \Phi \exp(-\Phi t)$$

$$\lambda(t) = N_0 \Phi \exp(-\Phi t)$$

失效强度函数

问题

- 固有错误数 N_0 如何求得？
- 比例常数 Φ 如何求得？



参数的点估计方法

- 模型输出中的某些值是无法直接得到的，最常见的比如残留缺陷数，这就要求模型给出一定的方法估计参数的值。
- 为了寻求这些参数的值，很自然地会想到用样本值x_1，x_2，…，x_n这组数据来估计参数的值，这个问题称为参数的点估计问题。

参数估计方法

- 极大似然法
  - 渐进一致性，渐进有效性，无偏性
- 最小二乘法
  - 当似然函数难以用解析法求解时常常使用
- 矩法
  - 用样本矩来估计相应的总体矩
- 贝叶斯估计

最大似然估计法简介

最常用的经典统计方法为最大似然估计方法。设母体X具有密度函数 $f(x;\theta)$ ，$\theta \in \Theta$

且 $x=(x_1,x_2,\cdots,x_n)$ 是来自X的一个给定的子样，则似然函数：

$$L(\theta,x)=\prod_{i=1}^{n} f(x_i;\theta)$$

是参数 θ 的函数。假如 $\hat{\theta}(x)$ 存在，则有： $L(\hat{\theta}(x);x)=\sup_{\theta\in\Theta} L(\theta;x)$

我们称 $\hat{\theta}(x)$ 为参数 θ 的最大似然估计。

若密度函数 $f(x;\theta)$ 关于 θ 是可微的，则 $\hat{\theta}$ 满足下列似然方程：

$$\frac{\partial}{\partial\theta_i} InL(\theta;x)\Big|_{\theta_i=\hat{\theta}_i}=0, \quad i=1,2,\cdots, \quad k.$$

1.4 参数的最大似然估计

- 如果在测试中测得失效间隔时间为 x_1，x_2，…，x_n，则参数 N_0 和 Φ 的似然函数为

$$L(N_0,\Phi)=\prod_{i=1}^{n}\Phi(N_0-i+1)\exp\{-\Phi(N_0-i+1)x_i\}$$

对上式两边同时取对数，得

$$\ln L(N_0,\Phi)=\sum_{i=1}^{n}(\ln\Phi(N_0-i+1)'-\Phi(N_0-i+1)x_i)$$

1.4 参数的最大似然估计

令

$$\frac{\partial \ln L(N_0, \Phi)}{\partial \Phi} = 0$$

$$\frac{\partial \ln L(N_0, \Phi)}{\partial N_0} = 0$$

得

$$\sum_{i=1}^{n} \frac{1}{N_0 - i + 1} - \sum_{i=1}^{n} \Phi x_i = 0$$

$$\frac{n}{\Phi} - \sum_{i=1}^{n} (N_0 - i + 1) x_i = 0$$

1.4 参数的最大似然估计

- 从上面两式可得

$$\sum_{i=1}^{n}\frac{1}{N_0-i+1}=\frac{n}{N_0-\frac{1}{t}\sum_{i=1}^{n}(i-1)x_i}$$

$$\Phi=\frac{n}{N_0 t-\sum_{i=1}^{n}(i-1)x_i}$$

- 其中 $t=\sum_{i=1}^{n}x_i$
- 利用数值计算法可以得到 N_0 和 Φ 的点估计值。

置信区间与置信水平

- 由数理统计方法所得的结论往往带有一定程度的不确定性，因此，常用置信水平来表示某一结论为“真”的概率。
- 设对于某一参数，通过抽取子样本对其推断出相应的统计量。如果有

$$P(t_1 \le \theta \le t_2) = r$$

其中 r 为置信水平，置信区间为 $[t_1, t_2]$

火龙果•整理
uml.org.cn

置信区间估计参数

- 预备知识

  最大似然估计量具有许多理想的最优性质：

  - 相合性

    当样本量趋于无限大时，估计量的方差趋于零，期望值趋于真实的总体参数。

  - 渐近正态性

    当样本量足够大时，估计量近似为正态分布。

渐近正态性

- 最大似然估计的一个重要优点是似然方程的解渐近于正态分布，设 θ_1 ，θ_2 为似然方程中的两个未知参数，则当n趋近于无穷大时，

$$\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix} \sim N\left(\begin{bmatrix}\theta_1\\ \theta_2\end{bmatrix}, \Sigma_{Cov}\right)$$

其中［θ_1，θ_2］是二维正态分布的均值矩阵，Σ_{Cov} 是二维正态分布的协方差矩阵，

置信区间估计

- Σ_{Cov} 是二维正态分布的协方差矩阵，即

$$\Sigma_{Cov} = \begin{bmatrix} V_{ar}(\theta_1) & C_{ov}(\theta_1,\theta_2) \\ C_{ov}(\theta_1,\theta_2) & V_{ar}(\theta_2) \end{bmatrix} \qquad \Sigma_{Cov} = \begin{bmatrix} r_{\theta_1\theta_1} & r_{\theta_1\theta_2} \\ r_{\theta_2\theta_1} & r_{\theta_2\theta_2} \end{bmatrix}^{-1}$$

Σ_{Cov} 可由下列各式算出：

$$r_{\theta_1\theta_1} = -E\frac{\partial^2 \ln L}{\partial\theta_1\partial\theta_1}$$

$$r_{\theta_1\theta_2} = r_{\theta_2\theta_1} = -E\frac{\partial^2 \ln L}{\partial\theta_1\partial\theta_2} \qquad r_{\theta_2\theta_2} = -E\frac{\partial^2 \ln L}{\partial\theta_2\partial\theta_2}$$

置信区间估计

$$a = r_{\theta_1\theta_1}$$

$$b = r_{\theta_1\theta_2} = r_{\theta_2\theta_1}$$

$$c = r_{\theta_2\theta_2}$$

$$V_{ar}(\theta_1) = \frac{c}{ac - b^2}$$

$$V_{ar}(\theta_2) = \frac{a}{ac - b^2}$$

由上面的式子求得各个参数的近似方差，则可以求得相应的置信区间。

置信区间估计

在J-M模型中，未知参数是 N_0 和 Φ ，从上述式中可以求得 $\hat{N}_0$ 和 $\hat{\Phi}$ 的方差、协方差和相关系数，则对于给定的 $\alpha(0<\alpha<1)$ 和置信度 $100\times(1-\alpha)\%$ ，参数 N_0 和 Φ 的置信区间分别为

$$\left(\hat{N}_0 - Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{N}_0)}, \hat{N}_0 + Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{N}_0)}\right)$$

$$\left(\hat{\Phi} - Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{\Phi})}, \hat{\Phi} + Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{\Phi})}\right)$$

置信区间估计

- N_0 和 Φ 的置信区间分别为

$$\left(\hat{N}_0 - Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{N}_0)}, \hat{N}_0 + Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{N}_0)}\right)$$

$$\left(\hat{\Phi} - Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{\Phi})}, \hat{\Phi} + Z_{1-\frac{\alpha}{2}}\sqrt{V_{ar}(\hat{\Phi})}\right)$$

- 其中 $Z_{1-\frac{\alpha}{2}}$ 是标准正态分布 $N(0, 1)$ 的 $1-\left(\frac{\alpha}{2}\right)$ 分位点。

参数的最大似然估计

- 在黄锡滋编著的《软件可靠性、安全性与质量保证》一书中，给出了如下结果：

$$V_{ar}(\hat{N}_0) = \frac{n}{A\Phi^2}$$

$$V_{ar}(\hat{\Phi}) = \sum_{i=1}^{n}\left(\frac{1}{N_0 - i + 1}\right)^2 \frac{1}{A}$$

$$C_{ov}(\hat{N}_0, \hat{\Phi}) = -\frac{t}{A}$$

$$\rho_{\hat{N}_0,\hat{\Phi}} = -\frac{t\Phi}{\left(n\sum_{i=1}^{n}\frac{1}{(N_0 - i + 1)^2}\right)^{\frac{1}{2}}}$$

$$A = \frac{n}{\Phi^2}\sum_{i=1}^{n}\frac{1}{(N_0 - i + 1)^2} - t^2$$

参数的最大似然估计

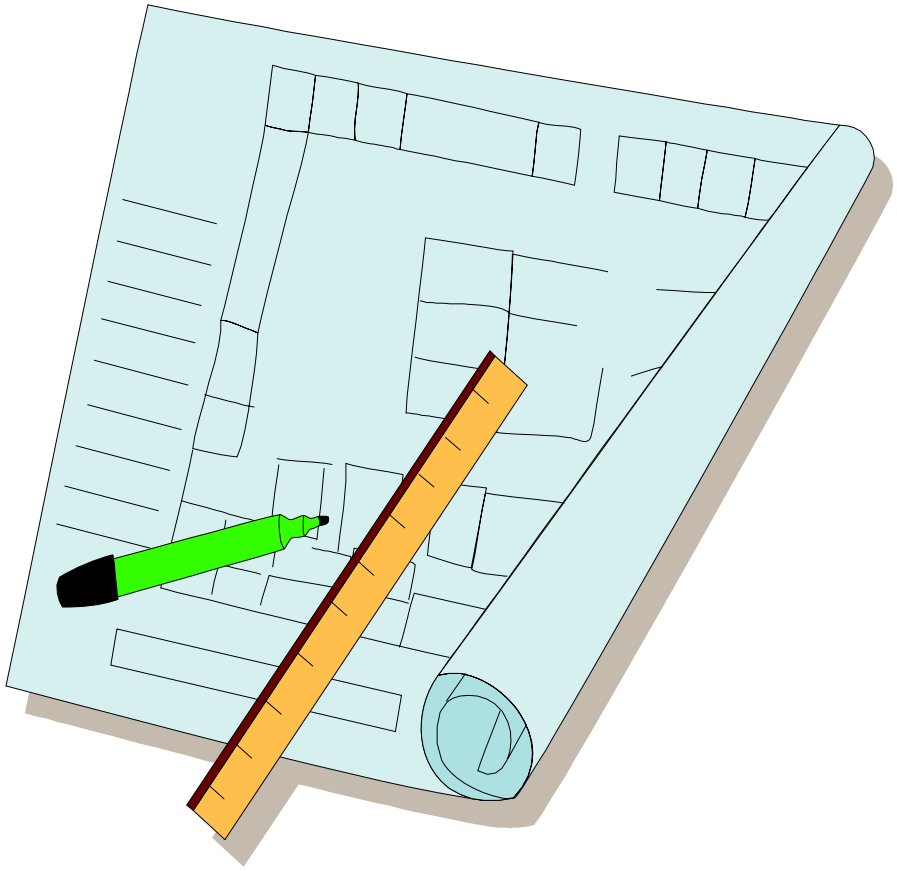
- 我们在研究中得到了如下结果：

$$V_{ar}(\hat{N}_0) = \frac{n}{n\sum_{i=1}^{n}\frac{1}{(N_0 - i + 1)^2} - \Phi^2\left(\sum_{i=1}^{n}\frac{1}{(N_0 - i + 1)\Phi}\right)^2}$$

$$V_{ar}\left(\hat{\Phi}\right) = \frac{\sum_{i=1}^{n}\frac{1}{\left(N_0 + 1 - i\right)^2}}{\frac{n}{\Phi^2}\sum_{i=1}^{n}\frac{1}{\left(N_0 + 1 - i\right)^2} - \left(\sum_{i=1}^{n}\frac{1}{\left(N_0 + 1 - i\right)\Phi}\right)^2}$$

课后作业

- 验证上述哪种结果是正确的。
- 要求给出完整的计算过程。



1.5 参数的最小二乘估计

模型参数 N_0 和 Φ 的估计值也可用最小二乘法求得。如果在测试中测得的失效时间为 x_1，x_2，…，x_n，令

$$S(N_0,\Phi)=\sum_{i=1}^{n}\left(x_i-\frac{1}{\Phi(N_0-i+1)}\right)^2$$

求偏导数$\partial S/\partial N_0$，$\partial S/\partial \Phi$，并令其等于零，可得

$$\Phi=\sum_{i=1}^{n}\frac{1}{(N_0-i+1)^2}\Big/\sum_{i=1}^{n}\frac{x_i}{N_0-i+1}$$

$$\left(\sum_{i=1}^{n}\frac{x_i}{(N_0-i+1)^2}\right)\left(\sum_{i=1}^{n}\frac{1}{(N_0-i+1)^2}\right)$$

$$=\left(\sum_{i=1}^{n}\frac{x_i}{(N_0-i+1)}\right)\left(\sum_{i=1}^{n}\frac{1}{(N_0-i+1)^3}\right)$$

从上式可解出参数的估计值 $\hat{N}_0$ 和 $\hat{\Phi}$ 。

1.6 可靠性参数预计

- 使用上述推导出的估计值，可相应地求得以下可靠性参数的估计值：
  (1) 失效强度
  (2) 失效数均值函数
  (3) 可靠度、不可靠度
  (4) 失效密度
  (5) *MTTF*
  (6) 残留缺陷数
  (7) 达到失效强度目标所期望的失效数
  (8) 达到给定可靠度所需的时间
  (9) 去除下*m*个缺陷所需的期望时间

(6) 残留缺陷数

$$N(x) = \hat{N} - i + 1 \qquad i = 1, \ldots, \hat{N}, t_{i-1} < x \le t_i$$

(7) 达到失效强度目标所期望的失效数

设程序的失效强度目标值为 λ_F，现在的失效强度为 λ_p，则达到失效强度目标值所期发生的失效数为：

$$\Delta u = (\lambda_P - \lambda_F) / \hat{\phi}$$

(8) 达到给定可靠度所需的时间

设R_m为给定的可靠度，则达到R_m所需的时间为：

$$x_{R_m} = \frac{-\ln R_m}{\hat{\phi}(\hat{N} - n)}$$

(9) 去除下m个缺陷所需的期望时间

若t_n时刻去除第n个缺陷，则去除下m个缺陷所需时间的期望值为：

$$T_m = \sum_{j=n}^{j=n+m} \frac{1}{\hat{\phi}(\hat{N} - j + 1)}$$

1.7 模型的极限条件

- 在用最大似然法和最小二乘法求参数估计值时，都得出了两个联立方程，这两个联立方程只能用数值法求解，因此需要选择一个初值，再根据初值输入后得出的结果重新选择数据逐次逼近。
- 在实际拟合数据时发现这些方程有时没法得到收敛的数值解，这个事实表明，用于模型拟合的数据应该满足某种条件，参数值才收敛，否则参数值发散。我们称这个条件为模型的极限条件。

1.7 模型的极限条件

最早研究模型极限条件的是Littlewood，他于1981年得出的J-M模型的极限条件为

$$\frac{\sum_{i=1}^{n}(i-1)x_i}{\sum_{i=1}^{n}(i-1)} > \frac{\sum_{i=1}^{n}x_i}{n}$$

1.7 模型的极限条件

用1.1中的实际数据对上式进行验证，经计算得出，式子左端＝12.33，右端＝9.6，满足模型极限条件。

$$\frac{\sum_{i=1}^{n}(i-1)x_i}{\sum_{i=1}^{n}(i-1)} > \frac{\sum_{i=1}^{n}x_i}{n}$$

1.8 FC型J-M模型

- 最初的J-M模型是以失效间隔时间TBF(Time Between Failure)作为考察对象，在使用模型时对数据的要求较严，需要记录每一次失效发生间隔时间。
- Sukert在1976年对J-M模型提出修正，使模型可用于失效计数FC(Failure Counting)型数据。FC型J-M模型所采用的假定与TBF型J-M模型的基本相同，仅有以下三点差别:

1.8 FC型J-M模型

(1)模型不再要求每个测试区间只发现一个错误。

(2)假定在第i个测试区间程序错误发生的强度函数为

$$Z(x_i) = \Phi(N_0 - M_{i-1})$$

$$M_i = \sum_{i=1}^{i} n_i$$

式中 N_0 ——指程序固有的错误数；

n_i ——在第 i 个测试区间检测到的错误数；

Φ ——比例常数；

x_i ——第 i 个检测区间的时间；

M_i ——第 i 个测试区间结束时检测的累积错误数。

1.8 FC型J-M模型

(3) 假定在第 i 个测试区间内可能检测到的错误 η_i 是随机变数，η_i 服从参数为$\Phi(N_0 - M_{i-1})x_i$ 的 Poisson 分布，即

$$P_r\{\eta_i = n_i\} = (\Phi(N_0 - M_{i-1})x_i)^{n_i}\exp\{-\Phi(N_0 - M_{i-1})x_i\}/n_i!$$

FC 型 J－M 模型的参数估计可用最大似然法求出，当错误数样本值为 n_1，n_2，…，n_m 时，其似然函数为

$$L(N_0,\Phi) = \prod_{i=1}^{m}(\Phi(N_0 - M_{i-1})x_i)^{n_i}\exp\{-\Phi(N_0 - M_{i-1})x_i\}/n_i!$$

参数 N_0 和 Φ 的点估计值 $\hat{N}_0$ 和 $\hat{\Phi}$ 可由下列两式中解出：

$$\sum_{i=1}^{m}\frac{n_i}{N_0 - M_{i-1}} = \frac{\left(\sum_{i=1}^{m}n_i\right)\left(\sum_{i=1}^{m}x_i\right)}{\sum_{i=1}^{m}(N_0 - M_{i-1})x_i}$$

$$\Phi = \sum_{i=1}^{m}n_i\Big/\sum_{i=1}^{m}(N_0 - M_{i-1})x_i$$

1.8 模型的极限条件

- FC型J-M模型的极限条件为

$$\left(\sum_{i=1}^{m} M_{i-1} x_i\right)\left(\sum_{i=1}^{m} n_i\right) \geqslant \left(\sum_{i=1}^{m} n_i M_{i-1}\right)\left(\sum_{i=1}^{m} n_i\right)$$

1.9 模型小结

- J-M模型是由Jelinski＆Moranda于1972年正式出版的。
- 最早提出且仍在使用，如应用在麦克唐奈道格拉斯海军工程。
- 为建模领域的工作提供了框架。

1.9 模型小结

- 这一模型是软件可靠性系统研究的真正开端，至今在软件可靠性建模方面具有基本意义。

原因何在呢？

1.9 模型小结

- 这一模型首次将硬件可靠性的基本概念，如可靠度、失效率、失效强度等，系统地引入软件可靠性领域。
- 这一模型确立了软件可靠性建模的黑箱方法，即不涉及软件内部结构，仅依据软件的外部行为（输入输出关系）刻划软件可靠性行为。
- 这一模型给出了软件可靠性建模的两个主要假设，一是测试用例的选取代表软件实际运行剖面，二是不同软件失效独立发生。
- 这一模型简单，众多的后续模型可视为其某种变形。

火龙果•整理
uml.org.cn

2.软件可靠性模型的理论简介

2.1 软件可靠性模型的组成

2.2 好的软件可靠性模型应具有的重要性质

2.3 软件可靠性模型的分类

火龙果•整理
uml.org.cn

2.1 软件可靠性模型的组成

2.1.1 模型假设

2.1.2 模型输出

2.1.3 参数估计方法

2.1.4 数据要求

火龙果•整理
uml.org.cn

2.1.1 模型假设

- **模型假设：**模型是实际情况的理想或简单化，总是包含若干假设。
- **标准假设：**
  - 软件运行方式和进行软件可靠性预计时的软件运行方式相同—确保预计结果的有效性

代表性假设　此假设认为软件测试用例的选取代表软件实际的运行剖面，甚至认为测试用例是独立随机地选取。该假设的实质是指可以用测试产生的软件可靠性数据预计运行阶段的软件可靠性行为。

火龙果•整理
uml.org.cn

2.1.1 模型假设

- **模型假设：**模型是实际情况的理想或简单化，总是包含若干假设。
- **标准假设：**
  - 在相同严重度类中，碰上每个缺陷的机会是相同的—保证同分布特性。

相同性假设　此假设认为所有软件失效的后果（等级）相同，即建模过程只考虑软件失效的具体发生时刻，不区分软件失效的性质。

火龙果•整理
uml.org.cn

2.1.1 模型假设

- **模型假设：**模型是实际情况的理想或简单化，总是包含若干假设。
- **标准假设：**
  - 当检测到缺陷时，各个失效是独立的—简化MLE推导

独立性假设　此假设认为软件失效独立发生于不同时刻，一个软件失效的发生不影响另一个软件失效的发生，譬如，假设相邻软件失效间隔时间构成一组独立随机变量，或假设一定时间内软件失效次数构成一个独立增量过程。

2.1.1 模型假设

- 特殊假设：
  - 各模型特有的

2.1.2 模型输出

- **模型输出：**软件可靠性模型常见的输出量有：可靠度、不可靠度、失效密度函数、失效强度、平均故障时间、残留缺陷数等。它们通常以数学表达式给出。

2.1.2 模型输出

- 可靠性估计
  - MTTF　　平均失效前时间
  - $R(x_0)$　　运行时间x_0时的可靠度
  - λ_P　　当前失效强度
- 可靠性预计
  - $\lambda_{\Delta t}$　　再测试Δt时间时的失效强度值
  - $\mu_{\Delta t}$　　再测试Δt时间时的失效数均值
  - $\Delta\mu_{\Delta t}$　　再测试Δt时间时所发生的失效数
  - $\Delta\mu_{\lambda F}$　　达到失效强度目标值所发生的失效数
  - $\Delta t_{\lambda F}$　　达到失效强度目标值还需进行的测试时间
  - $\Delta t_{\Delta\mu}$　　再发生Δμ个失效还需进行的测试时间

2.1.3 参数估计方法

- 参数估计方法：模型输出中的某些值是无法直接得到的，最常见的比如残留缺陷数，这就要求模型给出一定的方法估计参数的值。

2.1.3 参数估计方法

- 极大似然法
- 最小二乘法
- 矩法
- 贝叶斯估计

火龙果•整理
uml.org.cn

2.1.4 数据要求

- 一个软件可靠性模型要求一定的输入数据，即软件可靠性数据。软件可靠性数据是在软件可靠性测试或实际使用中观测到的软件失效数据。
- 在实际采集过程中，通常可以得到两类数据：失效间隔时间数据（完全失效数据）、失效计数数据（不完全失效数据）。
- 完全失效数据：每次失效时间数据，累计或间隔；实质上是指每一个失效都对应一个确切的时间；
- 不完全失效数据：一定观测时间内发生的失效数、等间隔或不等间隔观测时间；实质上是指单位时间内失效的次数。

失效时间数据

失效序号	失效间隔时间（小时）	累计失效时间
1	0.5	0.5
2	1.2	1.7
3	2.8	4.5
4	2.7	7.2
5	2.8	10.1
6	3.1	13.1
7	1.8	14.8
8	0.9	15.7
9	1.4	17.1
10	3.5	20.6
11	3.4	24.1
12	1.2	25.2
13	0.9	26.1
14	1.7	27.8
15	1.4	29.2
16	2.7	31.9
17	3.2	35.1
18	2.5	37.6
19	2.1	39.6
20	4.5	44.1
21	3.5	47.6
22	5.2	52.8
23	7.2	60.1
24	10.7	70.1

火龙果•整理
uml.org.cn

失效数数据

时间（小时）	失效数	累计失效数
8	4	4
16	4	8
24	3	11
32	5	16
40	3	19
48	2	21
56	1	22
64	1	23
72	1	24

火龙果•整理
uml.org.cn

2.2好的软件可靠性模型应具有的重要性质

a. 对于将来的故障行为能给出好的预测
b. 对有用的量能进行计算
c. 简单明了
d. 具有广泛的应用
e. 它应在合理的、与实际情况完全吻合的或十分接近的假设基础上做出

火龙果•整理
uml.org.cn

2.3 软件可靠性模型的分类

- 按随机性分类法：根据随机过程的假设，如过程的确定性或非确定性、马氏过程、泊松过程等，进行分类。
- 按软件出现的故障数进行分类：主要有错误计数模型和非计数模型，可数性或不可数性模型。
- 按模型参数的估计方法进行分类：主要有贝叶斯方法或非贝叶斯方法，最大似然估计法或最小二乘法，另外还有线性模型等。

2.3 软件可靠性模型的分类

- 按模型使用的时间方式分类：主要有日历时间和执行时间模型。
- 按修复过程分类：主要指强调对软件系统修复过程的一类模型，如：完全修复型的和不完全修复型的，完全排错型和不完全排错型的模型。
- 按对软件的内部结构是否了解进行分类：可分为黑箱模型和白箱模型。对它们的分类主要根据对软件的内部结构的了解程度以及对它们的结构能加以利用的程度来区分。

2.3 软件可靠性模型的分类

- 下图给出按随机性分类法对软件可靠性模型进行的分类。

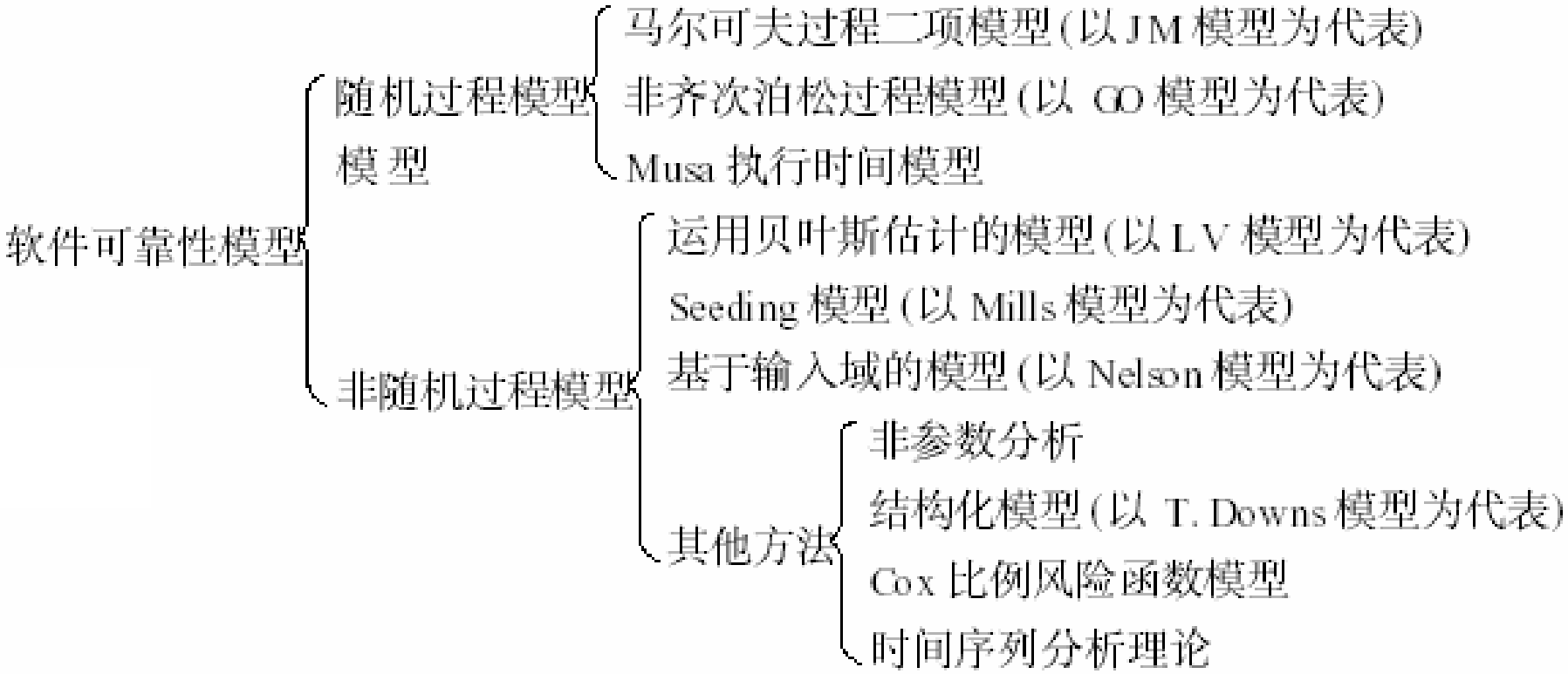


图 1　软件可靠性模型的分类

3.软件可靠性增长预计建模概况

- 先驱人物：Jelinski, Moranda, Shooman, 1972年
- 已有二百多个模型
- SRE中研究最多、最深入的领域
- 已进入应用阶段
- 但无普遍适用模型

3.软件可靠性增长预计建模概况

- **Jelinski-Moranda模型**
- **Jelinski＆Moranda于1972年正式出版他们的模型。这一模型是软件可靠性系统研究的真正开端，至今在软件可靠性建模方面具有基本意义，这是因为：**

①这一模型首次将硬件可靠性的基本概念，如可靠度、失效强度（率）等，系统地引入软件可靠性领域。

②这一模型确立了软件可靠性建模的黑箱方法，既不涉及软件内部结构，仅依据软件的外部行为（输入输出关系）刻划软件可靠性行为。

③这一模型给出了软件可靠性建模的两个主要假设，一是测试用例的选取代表软件实际运行剖面，二是不同软件失效独立发生。

④这一模型简单，众多的后续模型可视为其某种变形。

3.软件可靠性增长预计建模概况

- Halstead模型
- Halstead依据其‘软件科学’的思想于1972年提出此模型，试图利用软件复杂度确定软件缺陷数。此模型的重要性在于它可应用于软件开发的早期阶段（软件测试之前）。随着软件可靠性设计地位的提高，此模型的重要性也将显著增长。
- Littlewood-Verrall模型
- Littlewood＆Verrall与1973年发表此模型，其主导思想是将软件失效强度视为随机变量，从而开创了软件可靠性建模的bayes方法。此模型也间接考虑了发现的缺陷不被完全剔除的可能性

3.软件可靠性增长预计建模概况

- Musa执行时间模型
- Musa于1975年发表此模型，主要贡献在于提出将CPU执行时间作为软件可靠性的时间基准。
- Sukert模型比较工作
- Sukert于1976年给出其关于不同模型的分析比较的实验报告，开创了软件可靠性模型比较研究的先河。模型比较研究已成为建模研究的一个基本方面。

3.软件可靠性增长预计建模概况

- Nelson模型
- Nelson最早于1973年提出一个基于数据域模型，并于1978年得以完善。此模型是基于数据域模型的代表，成为目前应用最多的模型之一。
- Goel-Okumoto NHPP模型
- Goel&Okumoto于1987年发表此模型，将软件失效次数描述为非齐次泊松过程，是NHPP模型的代表。

3.软件可靠性增长预计建模概况

- Musa软件可靠性数据
- Musa于1978年发表了一批软件可靠性数据，对软件可靠性建模产生重要影响。软件可靠性建模研究的“标准”参考。
- 1983年，Yamada, Ohba和Osaki发表的一个NHPP模型，给出呈S一型的可靠性增长曲线的均值函数。
- 由K.Okumoto和J.D.Musa提出的对数泊松过程模型，具有一个初始错误率λ，以及对应于λ的一个递减率。模型的日历时间部分则与执行时间模型的日历时间部分相同。

3.软件可靠性增长预计建模概况

- Musa－Iannino－Okumoto著作
- 1987年发表“软件可靠性”一书。首次对众多软件可靠性模型给出统一数学描述，1999年John Musa又出版了《软件可靠性工程》一书，该书是一本权威性著作，最近该书又作了完善，出了第二版。
- Cai模糊模型
- Cai于1990年提出模糊软件可靠性模型，试图以模糊方法更本质的刻画软件可靠性行为，对软件可靠性建模研究提出挑战。
- T. Bendell
- T. Bendell将试探性数据分析法（EDA）引入软件可靠性评估，且使用时间序列分析法以及成比例的风险函数建立软件可靠性模型

3.软件可靠性增长预计建模概况

- Y.Tohma, K.Tokunaga, S.Nagase和Y.Murata
- 1989年，Y.Tohma, K.Tokunaga, S.Nagase和Y.Murata提出一个新的，以超几何分布为基础的模型，用于估计软件中剩余错误个数。
- Tsu-Feng Ho, Wah-Chun Chan 和Chyan-Goei Chung
- Tsu-Feng Ho, Wah-Chun Chan 和Chyan-Goei Chung提出一个模块结构模型，通过每个模块的可靠性来估计软件系统的可靠性。还提出结合信息论的方法，建立软件可靠性模型的方法。

3.软件可靠性增长预计建模概况

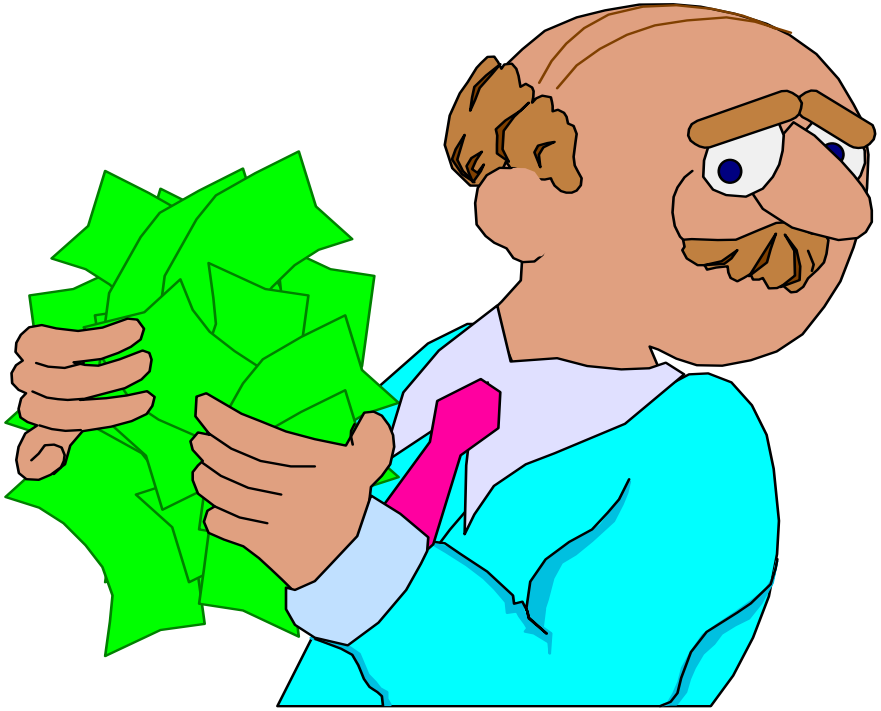
- K.W.Miller
- K.W.Miller等人，开发出以软件的黑盒子为基础的理论分析方法，解决：
  - 随机测试过程中，当观察到的故障次数为零时，如何估计当前版本的故障概率
  - 使用分布（use distribution）与测试分布（test distribution）不匹配时，对估计的故障率的调整
  - 估计故障概率时，将随机测试的结果与其它的信息结合起来分析

3.软件可靠性增长预计建模概况

- N.Karunanith, Y.Malaiya和D.Whitley应用神经网络系统理论预测软件可靠性。利用向前神经网络（a feed-forward neural network ）对三个数据集合进行软件可靠性估测，结果发现：
  - 神经网络方法在软件可靠性估测中显示出良好的一致性，这是一般现有软件可靠性模型无法做到的。
  - 由神经网络方法估计出的软件中的错误个数，始终较使用现有其它软件可靠性模型所估计出的结果要少。
  - 神经网络方法的估测能力，可通过由进行分析的数据，与其它的模型进行比较获知。

思考

- 你觉得模型的假设存在哪些问题呢？

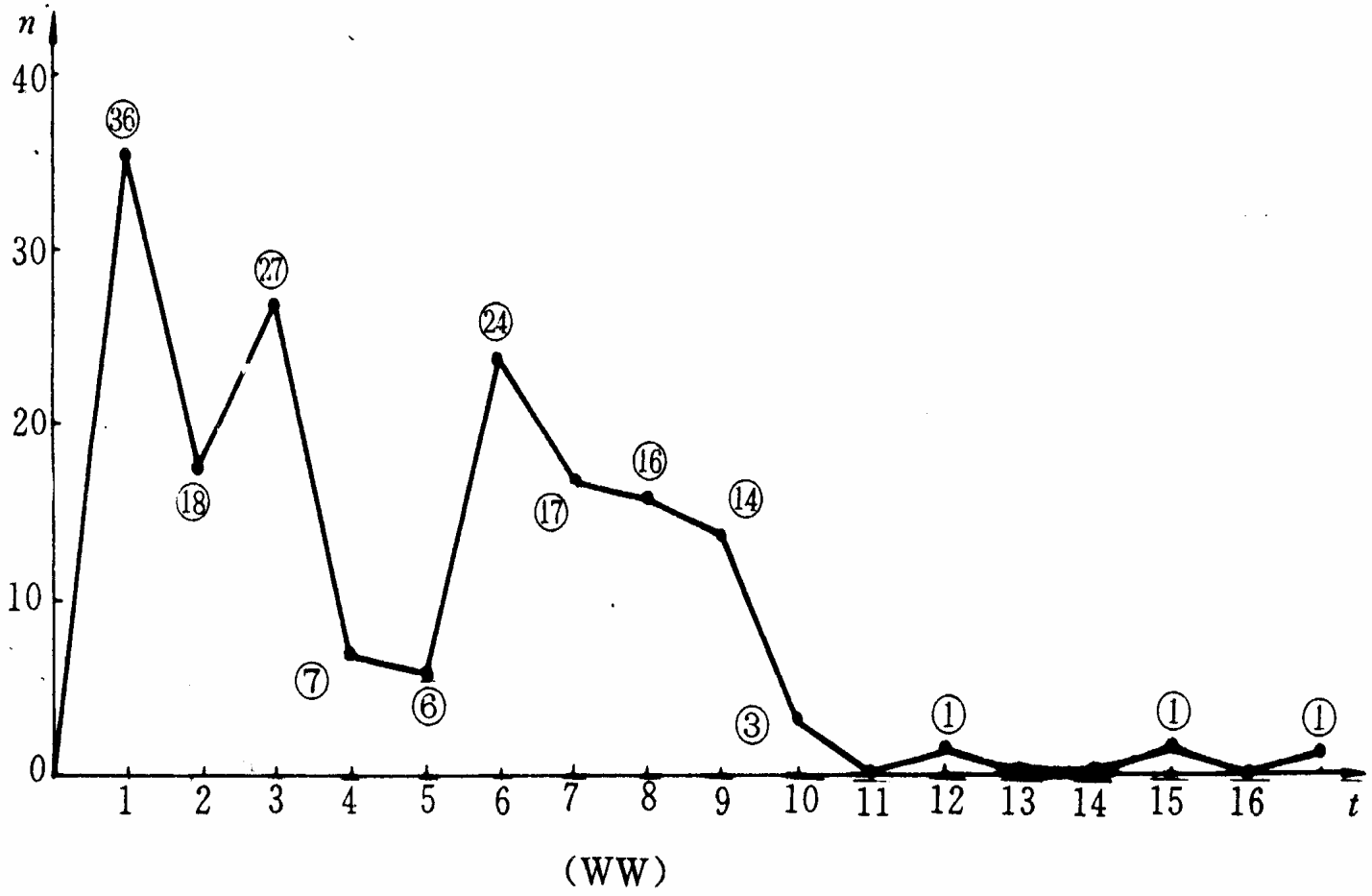


模型假设存在的问题

- 从许多假设来看，有很多是与软件开发实际不相符合的，许多软件工程师与软件管理人员无法接受。
- 许多现存模型(特别是那些早期的软件可靠性模型)，考虑到排错引入新的错误会使问题复杂化，于是假设排错不引入新的错误。这样做的结果虽然使理论上的处理简单了，但与实际情况相距太远。

模型假设存在的问题

- 软件的开发靠人完成，则排错问题要人完成，人类行为的不可预测性无论在开发还是排错，同样要表现出来。事实上，由于排错时的某些处置失当，往往会产生许多副作用，引入一些始料不及的新错误，是十分自然的。这也正好解释了我们在对软件中出现的错误进行观察记录时，为什么经常会大幅度地振荡的原因。
- 引入新错，另一方面的原因还在于软件产品各模块(指结构化的软件产品而言)间的逻辑关系错综复杂、互为因果，故而使得局部的某些改动甚至可能产生牵涉全局性的许多问题。
- 随后的一些模型，虽然允许可以引入的错误数为1、2或其它，正是由于看到了这一问题，才作了一些调整变动，但仍未能从根本上解决问题。


n
40
30
20
10
0
36
18
27
7
6
24
17
16
14
3
1
1
1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 t
(WW)

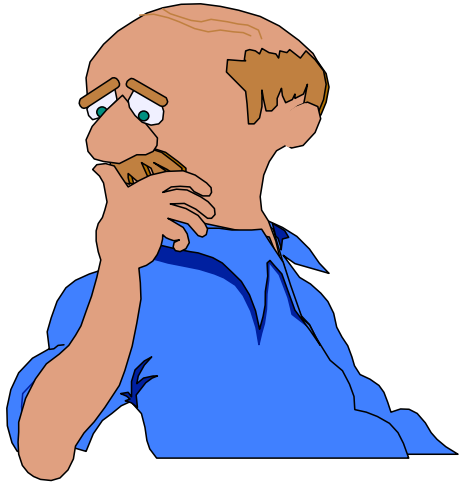
模型存在的问题

- 关于测试时的输入空间“覆盖”使用空间的假设，也是不现实的。为了尽量保证测试能充分反映出软件产品将来所有可能的使用情况，有必要在设计测试的数据和条件时，使它们能尽可能地反映出软件产品将来使用时的情况、条件和环境。但这并不意味着测试就一定是完全的。测试时的输入空间充其量也只能是从使用时的输入集合的全集合中选取的子集合。如果使用时的问题空间是无穷集合(大多数实际情况也正是如此)，则这样选出的子集合，无论如何也不可能“覆盖”问题空间。假如一定要做到“覆盖”，只有使测试无限制地进行下去，而这也正是我们力图要避免的。
- 由"覆盖"问题引伸出的测试环境与使用运行环境一致的问题，也是同样性质的，这一假设也是人们一种良好愿望的反映。特别对于那些包括软件的无法进行试验的系统，则这一假设更显得不合实际。

模型存在的问题

- 不同软件产品的开发过程，由于参加者不同，他们各自的训练、业务经历、程序设计风格都不相同。因此，通过他们各自的大脑思维所产生出来的"逻辑产品"，个性多于共性，这是一种必然现象，也正是软件工程所面临的一大难题。
- 模型假设的局限性太多，势必影响到它们的应用范围。目前，软件工程界对于软件可靠性模型的诸多疑虑，也多半来自于此。如何改进它们，是软件可靠性今后理论研究的重大课题之一。它的突破，也就会消除软件工程界的疑虑，使软件可靠性理论得到更广泛的应用，从而，必然反过来又促进软件可靠性理论的发展。

- 软件可靠性模型没有普适性，一个模型可能仅对一个或几个软件做出较为准确的评估和预计，因此，如何选择、评价模型就成为一个很有必要研究的课题。

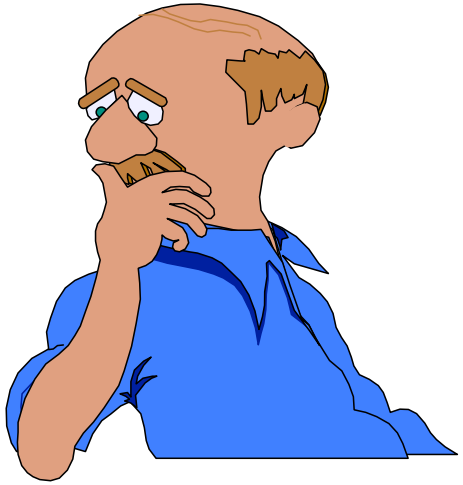


软件可靠性模型的选取原则

- 通常认为，评价软件可靠性模型对一个给定项目的支持时应使用下列准则。

(1) 预计有效性：每个模型的预计质量方面的性能和正确性。在这方面的度量是，准确性、趋势、偏移和噪声。

(2) 容易进行参数测定：测定每个模型的参数所产生的资源需求和影响，即：模型所需要的参数数量以及估计这些参数的难度。



软件可靠性模型的选取原则

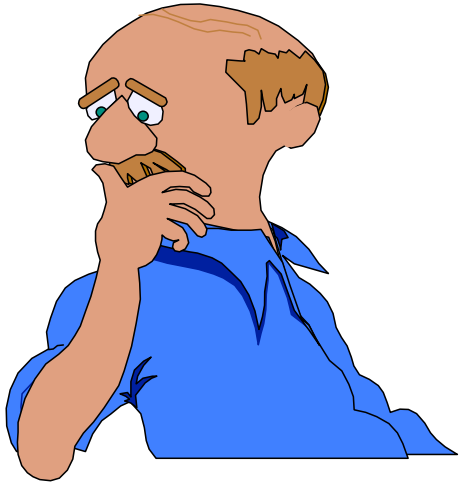
（3）假设的质量：该准则是指假设与真实情况的接近程度，以及对特殊环境的适应性。

（4） 能力：能力指模型对与可靠性有关的量的估计能力如何。

（5）适用性：对软件在测试和运行环境中的演变和修改的处理能力。

（6）简单性：模型建模原理、数据采集、程序实现和确认的容易程度。

（7）对噪声的不敏感性：尽管在输入数据和参数中有小的差别，模型仍能产生结果，同时对显著差别又不丢失相应的能力



总结

- 以JELINSKI-MORANDA模型为例，对软件可靠性增长预计模型的研究方法进行了学习。
- 软件可靠性模型的理论简介。
- 软件可靠性增长预计建模概况。

火龙果•整理
uml.org.cn