计算机的可靠率与容错计算技术的应用

电子计算机专业 陈光熙

引 言

随着计算机制造工业的发展及其应用领域的日渐扩张，对机器的“可靠性”或“稳定时间”就有了愈来愈高的要求。此事虽小，但亦关系到工农业生产、国防与科学技术的现代化。在这里，同在其他领域一样，也迫切地需要赶超世界先进水平。

什么叫做可靠性，什么叫做稳定时间呢？可靠性就是完成任务的把握；说可靠性有多高，就是说完成任务的把握有多大。看来，可靠性应当以任务的完成率或百分比来衡量。不言而喻，计算机需要完成的任务——算题的任务或控制的任务等——愈庞大、愈复杂、愈经久，能完成它的把握就愈小，可靠率就愈低。反过来说：愈是要求以更大的把握即更高的可靠率来完成任务，该任务本身就只能是愈简单、愈短暂的。

要求计算机“可靠地运行”或“长期稳定”是一种过于绝对化的说法。因为计算机是集合大量的原器件通过大量的工艺操作而制成的，它要在每秒时间内完成数十万、数百万乃至数千万次基本操作，而这种操作又在复杂的“程序”控制下进行的，其可靠性的高低或稳定时间的长短符合于大数量事件发生的统计性规律，换句话说，用概率论的算法来估算计算机系统的可靠率或“平均无故障运行时间”是比较合适的，也许是唯一可能的。

对于一个具体规模的计算机系统而言，完成任务的概率决定于完成该任务所需的时间；时间愈长，无故障运行的概率愈低；要求在运行中不出故障（不出错）的概率愈高，即要求完成任务的把握愈大，该任务的延续时间只能是愈短的。

本文面向非专业读者，对计算机的可靠率、稳定时间等词，介绍一些符合于概率论的，计量的即数值的概念。然后讨论提高可靠率或稳定时间的途径，从而引进宽裕技术（冗余技术）的概念，采用宽裕技术的范围包括硬件、软件、时间各个方面。就是说在以上各个方面可以采用多于必不可少的数量来提高可靠率、无故障运行时间以及所谓可用率即机器的无故障运行时间占总时间的比值。

限于笔者水平，错误在所难免，敬希读者予以指正。

1. 可靠率的数学概念

人们通常谈到，一台计算机的可靠性和稳定性。这样的名词只有质的概念而没有量的概念，是难以明确所称的可靠性是什么东西的。如果把可靠性改称可靠率，就似乎包

含了量的概念，可靠率总是个百分数。

为了进一步明确可靠率的含义，我们可以认为：一台计算机、或计算机的一个部件，在指定的时间间隔 t 内，无故障运行的概率，就是它在这段时间内的可靠率 R。这样看来，可靠率是时间 t 的函数。

$$R = f(t) \tag{1}$$

$f(t)$ 的具体形式不难推导出来：首先，时间 t 愈长，$f(t)$ 必然愈小，这是显而易见的。其次，在时间 t 极小而趋于零时，$f(t)$ 愈高而以 1，即百分之百为极限：$f(o) = 1$。第三，不论时间多么长，$f(t)$ 总没有理由成为负数。从这三点可以看出代表(1)的曲线，应经过(0,1)点，同时是单调的下降曲线，而且全部位于第一象限，如图 1。第四，让我们研究一下，时间由 t 增到 $t+\Delta t$ 时，$f(t+\Delta t)$ 与 $f(t)$ 的关系。因为 $f(t+\Delta t)$ 是在 $0<t\leqslant t+\Delta t$ 这段时间内无故障运行的概率。在这一段时间内，无故障运行，就既要在 $(0、t)$ 时间内无故障运行，又要在 $(t+\Delta t)$ 时间内无故障运行，这是概率论的要求。因此，

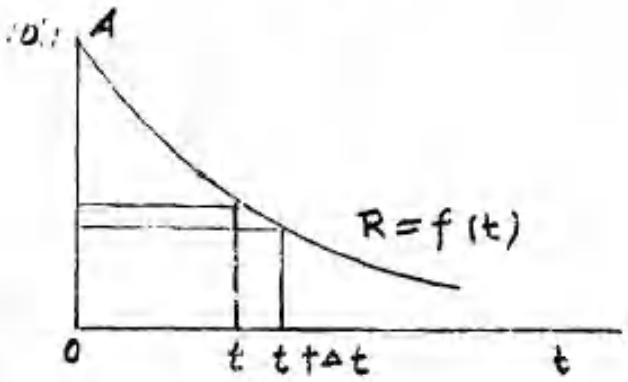


图 1 可靠率曲线的概貌

$$f(t+\Delta t) = f(t) \cdot f(\Delta t)$$

从而

$$f(t+\Delta t) - f(t) = f(t) \cdot f(\Delta t) - f(t)$$

$$f(t+\Delta t) - t(t) = f(t)[f(\Delta t) - 1]$$

$$\frac{f(t+\Delta t) - f(t)}{\Delta t} = f(t) \cdot \frac{[f(\Delta t) - 1]}{\Delta t}$$

当我们考虑愈来愈小的 Δt 时，左端的极限为 $f'(t)$，"′" 代表 $\frac{d}{dt}$，即时变率。右端的极限应为 $f(t) \cdot f'(0)$；所以

$$f'(t) = f(t) f'(0) \tag{2}$$

这里 $f'(0)$ 是个与 t 无关的数，即对 t 来说是个常数。从 (2) 得

$$\frac{f'(t)}{f(t)} = f'(0)$$

积分后 $\quad l_n f(t) = t f'(0) + C$

但我们已知 $\quad t = 0$ 时，$\quad t(t) = 1$，所以

$$C = 0$$

$$l_n f(t) = t f'(0)$$

$$f(t) = e^{t f'(0)}$$

如图 2，连曲线的(0.1)点作切线 AB 与 Ot 轴交于 B 点，命 $OB=\tau$，于是

$$f'(O)=-\frac{OA}{OB}=-\frac{1}{\tau}$$

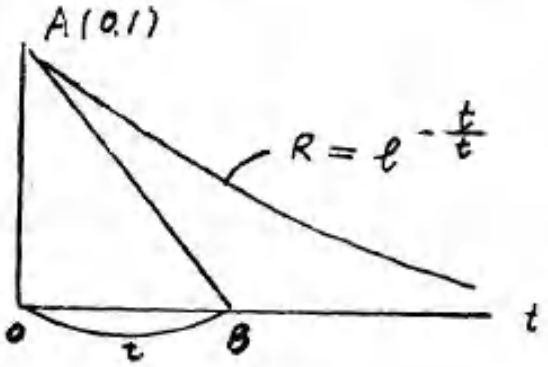


图 2 可靠率曲线

最后，我们得：

$$R=e^{-\frac{t}{\tau}} \tag{3}$$

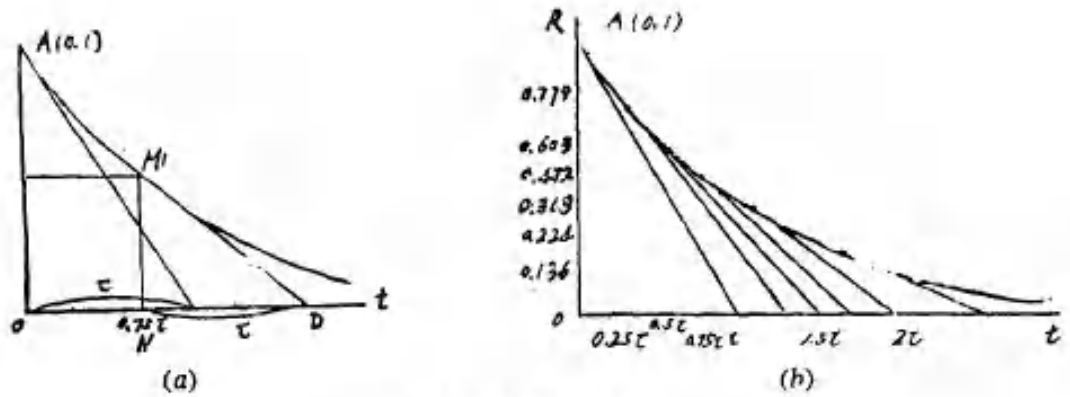
这就是可靠率 R 作为时间函数的表达式，它说明可靠率服从负指数分布规律。

曲线（3）的几何性质。

(a) 曲线(3)是一条仅仅包含一个参数 τ 的曲线，对不同的 τ 值，该曲线只有尺寸上的不同，而无性质上的不同，好比圆只有一种圆，抛物线只有一种抛物线，不象椭圆那样，因为具有两个参数，所以不仅有大小的区别，还有形状上的区别。

b) 由 (2) 知

$$f'(t)=f(t)\cdot f'(O)$$
$$=-\frac{1}{\tau}\cdot f(t) \tag{4}$$



(a) (b)

图 3 曲线（3）的切线

利用(4)易于绘出各点的切线，例如 $t=0.75\tau$ 时，$f(t)=0.472$。由 $N(0.75)$ 点向右取 $ND=\tau$，$MD=$ 即为 M 点的切线，如图 3 (a)。

照此法绘出 $(0.25\tau,\ 0.725)$ ……各点的切线，就能比较精确地绘出曲线 (3) 如图 3(b)。

c) 平均稳定时间

如前所述，曲线 (3) $R=e^{-\frac{t}{\tau}}$ 上任意一点的横座标表示无故障运动时间，而其纵座标则表明能完成这一无故障运行的概率，在适当处作直线 BC 平行于 OA，使面

积ACE等于曲线以下，Ot以上，BE以右的面积，如图（4）。则DE应为横座标的平均长度，即相对于各种不同的可靠率而言，t的平均值$T_{均}$，由此可见。

$$T_{均}\cdot OA = T_{均} = \int_0^{\infty} e^{-\frac{t}{\tau}}\,dt = -\tau e^{-\frac{t}{\tau}}\Big]_0^{\infty}$$

$$= \tau \qquad (5)$$

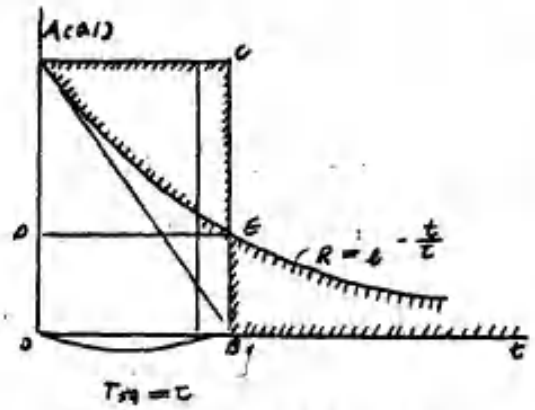


图 4 平均无故障运行时间τ

即平均稳定时间正好是曲线（3）的参数τ，而达到这平均稳定时间的概率只不过是

$$e^{-1} = 0.368 \quad \text{如图 4}$$

平均稳定时间τ是衡量计算机可靠性的一个重要指标，因为对具有不同τ值τ_1、τ_2，的两台机器而言，在各种相同的可靠率要求下，其无故障运行时间与τ、τ_2，成比例如图 5，

例如有两台机器，第一台的平均稳定时间为τ_1，第二台的平均稳定时间为$\tau_2 = 2\tau_1$，那么以 90% 的可靠率的无故障运行时间分别为：$0.105\tau_1$与$0.105\times 2\tau_1$，即以 90% 的可靠率的无故障运行时间也成 1 与 2 之比。

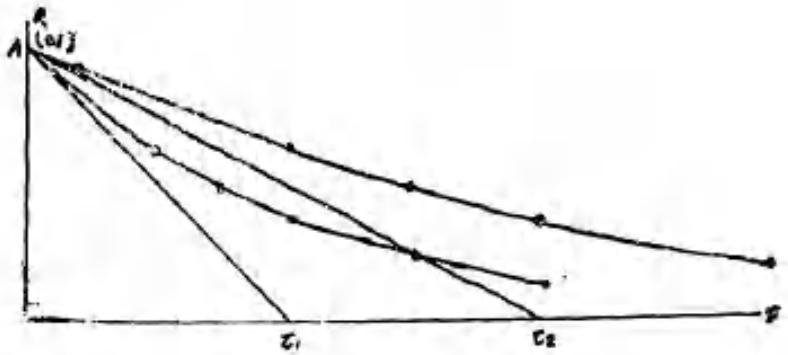


图 5 对不同τ值的机器，以相同的可靠率无故障运行时间与τ值成比例

2. 计算机各组成部分可靠率的计算

公式（3）可应用于整机，亦可应用于机器的任何组成部分，不过各有其不同的τ值而已。比方说，整机由运算控制器，内存与有关输出/入设备组成。
但由于机器在运行中，以上三者缺一不可，这三部分的可靠率成“与”的关系来共同保证整机的可靠率：

$$R_{整机} = R_{运控}\cdot R_{内存}\cdot R_{外设} \qquad (6)$$

此处有关外设，不包括闲置不用的外设，考虑到（3）

并设$\frac{1}{\tau}=\lambda$，得

$$e^{-\lambda t}_{整机}=e^{-\lambda t}_{运控}\cdot e^{-\lambda t}_{内存}\cdot e^{-\lambda t}_{外设}$$

今以λ、λ_1、λ_2、λ_3分别代表上式中的各个λ值，于是

$$e^{-\lambda t}=e^{-\lambda_1 t}\cdot e^{-\lambda_2 t}\cdot e^{-\lambda_3 t}$$

$$=e^{-(\lambda_1-\lambda_2+\lambda_3)t}$$

$$\lambda=\lambda_1+\lambda_2+\lambda_3 \qquad (7)$$

τ既然是平均稳定时间可以小时计，就是说起动后平均τ小时才遇到故障，也就是说平均故障率为1次每τ小时，或τ次每小时，因此λ/小时是每小时的平均故障率，由此可见(7)的函义无非是整机的平均故障率λ/小时为各组成部分的平均故障率之和：

$$\lambda_1/小时+\lambda_2/小时+\lambda_3/小时$$

基于相同的理由，我们可以得出：

$$\lambda_{主机}=\lambda_{运控}+\lambda_{内存}$$

$$\lambda_{运控}=\lambda_{机架}+\lambda_{扦件}$$

$$\lambda_{内存}=\lambda_{机架}+\lambda_{扦件}+\lambda_{存贮体},$$

$$\lambda_{机架}=\lambda_{扦座接触}+\lambda_{焊点}+\lambda_{导线},$$

$$\lambda_{扦件}=\lambda_{扦件_1}+\lambda_{扦件_2}+\cdots\cdots$$

对于n个相同的扦件，可以写

$$\lambda_{n个扦件}=n\cdot\lambda_{每扦件}$$

又，扦件是由集成电路n_1个组件，n_2个焊点等所组成，所以大致有：

$$\lambda_{扦件}=n_1\lambda_{组件}+n_2\lambda_{焊点}+n_3\lambda_{化孔}+n_4\lambda_{扦头接触}$$

λ组件，组件故障率的失效率，要看原器件产品质量及筛选老化规程的严格程度，一般可以达到$5\sim2\times10^{-7}$/小时。

λ焊点，λ化孔，λ扦头接触关系到工艺水平，一般可以达到$10^{-8}\sim10^{-9}$/小时，但在很大程度上依赖于整机制作过程的严格要求与认真的质量检查。

不难看出，一台机器总的故障率λ是各组成部分的大量的（几十万到几百万）故障率的和，因此机器愈庞大愈复杂，其故障率愈高，亦即是它的平均稳定时间愈短。

3. 提高可靠性的途径

(a) 常规设计计算机。这种机器不论它规模有多大，原器件、焊接点等有多么多，但没有一处可以有故障而不引起全机的故障的，到目前为止已运行的计算机一般都是属于这种设计。即使加一些备分扦件，也只能有时缩短一些修复时间，对于这种机器提高其可靠的手段不外通常所强调的一些办法，在逻辑设计中注意可靠性，在时间上留有余量，对原器件加强老化筛选以提高（λ原器件），在工艺上严格把关等，应当认识到，这些手段是非常重要的，不论如何强调这一点也不致过分，但采取这样一种途径代价亦很昂贵而且很快就到尽头。例如对原器件说，为了达到5.10^{-7}/小时已经要淘汰

掉50%的器件，如果更要达到2.10^{-7}小时，则必须淘汰75%，这表示着在原器件方面的造价成倍的增加，又如对金属化孔，或焊接点，如果把它们的电阻值限制在较低的门槛值以下，稍超过，就认为不合格，则必然要淘汰大量的扦件板制品，从而使机器的成本从这方面大为提高等。

另外，由于这种机器的可靠率曲线，即曲线（3），只有一种形式，用它的平均稳定时间τ，就足以完全衡量它对各种不同任务时间的可靠率，换句话说，如果通过付出代价将τ提高了20%，即在可靠率为0.368时，无故障运行平均时间提高了20%，那么以90%的可靠率无故障运行平均时间也只能提高20%，参考图5。

(b) 非常规设计的计算机，宽裕术平或容疵计算技术，先从一个例子说起，例如有相同的三台机器，其输出分别为f_1 f_2 f_3，我们使f_1 f_2 f_3，通过或门如图6，命总的输出将为F。f_1、f_2、f_3中至少有两个为1则F为1，f_1 f_2 f_3中至少有两个为0时F为O，这种F对f_1 f_2 f_3的依存关系称作三中取二。

f_1	f_2	f_3	F
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	1
0	0	1	0
1	0	1	1
0	1	1	1
1	1	1	1

图7 F的真值表

与或门起着一种“表决”作用。设每机的可靠率为R

$$R = e^{-\frac{t}{\tau}}$$

通过表决后总的可靠率R_3，将为：

$$R_3 = R^3 + 3R^2(1-R)$$
$$= 3R^2 - 2R^3$$

或：
$$R_3(t) = 3e^{-\frac{2t}{\tau}} - 2e^{-\frac{2t}{\tau}} \qquad (9)$$

方程（9）一点一点绘出来就成图7中曲线（9）的形状，

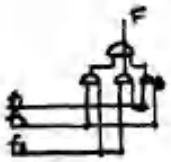


图 6 三中取二表决逻辑

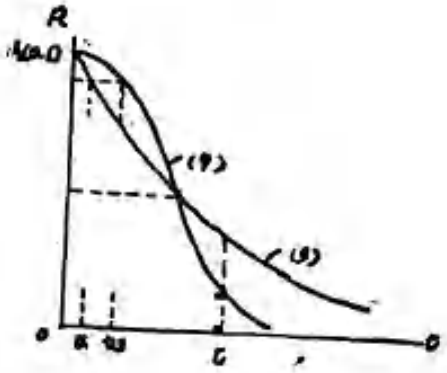


图 7 三模表决系统可靠曲线

下面讨论一下将三台一样的机器联结成三中取二表决系统有什么优点，或在什么条件下有优点：

首先计算一下它的平均稳定时间$MTBF$

$$(MTBF)3 = \int_0^\infty \left(3e^{-\frac{2t}{\tau}} - 2e^{-\frac{3t}{t}}\right)dt$$
$$= \left[-\frac{3}{2}\tau e^{\frac{2t}{\tau}} + \frac{2}{3}\tau e^{-\frac{3t}{\tau}}\right]_0^\infty = \frac{5}{6}\tau$$

这就是说，用了三倍多的器材，费了三倍多的生产工程，反而得到一个小于一台单机的平均稳定时间，所以从平均稳定时间的角度来看，三模表决系统是不可取的。

其次，从图7中亦可以看到，当可靠率超过0.5时，即$t<0.693\tau$时，$R_3(t)>R$，但当$t>0.693\tau$时，$R_3(t)<R$，这就是说：当$t>0.693\tau$时，三模表决系统是不可取的（多数并不代表正确性），而在$t<0.63\tau$时，尤其是在$t\ll0.63\tau$时，三模表决系统变成是极为可取的。

对宇宙飞船上所载的计算机来说不但要求体积小，重量低，功耗量限于太阳能所能供给能量，还要求稳定时间长，所谓稳定时间长不是平均稳定时间长(可靠率=0.368)而是在高可靠率条件下的稳定时间长，例如要求可靠不低于90%，则从图7上看：

单机的无故障运行时间：$t_1\approx0.1\tau$

三模表决系统的无故障运行时间：$t_2\approx0.25\sim0.3\tau$就是说，在无故障运行时间得到2.5～3倍的提高。

取$R_3(t)$的导数得

$$R_3'(t)=6R-6R^2$$

$t=0$时，　　$R(O)=0$

这说明在A点曲线(9)的切线是水平的，因此对可靠率的要求愈高时，t_2/t_1的比值愈高，愈显示三模表决系统的优越性。

再取$R(t)$的二次导数，得

$$R_3''(t)=6-12R$$

$R_3''(t)=0$时　　$R(t)=0.5$　　$t=0.693$

即曲线（3）与曲线（9）的交点（0.693，0.5）也是曲线（9）的转折点，这更进一步说明在这点以左、三模表决系统（9）对单机系统（3）有优越性。

必须试问一下这优越性是那儿来的，是通过付出什么代价取得的，所付出的代价是增添了两倍（实际上是两倍还多一点）的原器件与生产工作量。（3）是常规设计，可靠率服从负指数分配规律，一个器件或一个焊点的故障就导致机器的故障。（9）是在（8）的基础上增添了东西，一个器件或焊点的瞬间故障不再导致整机的故障。换句话说整机能够容忍一些故障。所以三模表决系统代表着一种“故障容忍计算机”而按常规设计的机器（3）则是不能容忍任何故障的。

故障容忍计算机，通常简称为容错计算机，或者更恰当地称为容疵计算机，值得注意是故障容忍计算机的故障分布规律已不是象常规设计计算机那样僵硬地服从于负指数分配规律，而可以有种种不同的随着设计而变化的分布规律。

上面说三模表决系统之所以能够容忍故障是由于在它的设计中使用了多于常规设计所必需的东西，所以容错计算机所采用的技术亦称为“冗余技术”或者更恰当地称为“宽裕技术”。

4. 计算机系统的类型

容疵技术或宽裕技术包括以下各个方面，即硬件上的宽裕，信息上的宽裕，时间上的

宽裕（指令、复执、程序卷回）以及软件上的宽裕如故障诊断等。

在上节所举的例子中，应用容错技术，对短的任务时间可以提高设备的可靠率或者可靠率要求极高时，可以延长任务时间；但对平均无故障运行时间反而不利。是不是一切容错技术的应用都导致同样的结果呢，当然不是的。

采用宽裕技术，并不是多多益善，采用多少才是恰当，要看对机器性能的具体要求而定，从这个角度看，计算机系统大体上可以区别为两类：高度可靠的计算机系统与高信息通量的计算机系统即通用机类型的系统。

1）高度可靠的计算机系统，这类系统的主要性能指标就是非常长的无故障运行时间，但并不要求有多大的信息通量，即单位时间内计算能力不一定要求有多么大，这一类机器还可以分为两种：无人维修的计算机系统与有人工维修的系统。

1.1）无人维修的系统，这种计算机系统，用之于无人环境，比方说用在星际航行的宇宙飞船中，为了保证长期的，无人维修的，无故障运行，在系统中设置贮备次系统或贮备部件，而贮备又分为协同操作的贮备，休止待命的贮备与协同、待命二者混合的贮备。

1.11）协同操作贮备，前面所举的例子，三机协同操作，在文献中称为三模宽裕（*TMR*），〔1〕，这是协同操作系统的最简形式，它仅能排除每个次系统的瞬间故障而不能应付永久性故障，为了进一步提高可靠率，延长无故障运行时间，新的发展是采用更多的贮备成为 n 模宽裕其工作原理是 n 中取 $n-1$。在 n 份次系统中，如果有一份的输出与其他 $n-1$ 份的输出不一致，就将它排除，以剩下的 $n-1$ 份的输出作为系统的输出。如果不一致只是瞬间的，在被排除的次系统恢复正常后还可以让它归队。如果是永久性的不一致，就永久将它排除在系统之外。随着时间的推移，系统中次系统的份数逐渐下降：

$$n \to n-1 \to n-2 \to \cdots 3$$

最后退化为三模宽裕系统。〔2〕

1.12）待命贮备，在从事研究宽裕技术的专门家队伍中，有一部分人认为处于休止状态的部件，即接通信息线而不接通电源的部件，其寿命应较长于工作中的部件。以这种假设为依据，在系统中设置 N 份相同的次系统，仅仅让一份工作，其他份则处于待命接力的状态。这样一计算，整个系统的寿命要比 N 份协同操作的系统要长得多，〔3〕。但这里存在着一些不好解决的问题：一个次系统的无故障运行时间的离散率很大，不能采取定期切换的方式，一个次系统由休止状态突然转入工作状态叫作冷起动，一般认为冷起动的成功率是比较低的，必须给它一段操练时间，称为起动前的“教育”时间，应及时开始进行教育，才能使接力运行不致于脱节，而且是经济的，即不是过早地淘汰原来在工作中的次系统？在这方面还未看到什么经验的介绍。

1.13）混合贮备，以三模宽裕为核心，再配上 $n-3$ 份待命贮备这叫作混合贮备，在这方面文献中材料比较多，就中以 AVizennis 为主，介绍了比较繁复的数学模型，其主导思想是当作为核心的三模宽裕的三份次系统中，遇到有一份失灵时，就让待命中的 $n-3$ 份之一去接替它。当然，这里仍旧存在着一个教育新兵实现切换接替的成功率问题。

1.2) 有人工维修的计算机系统，人的因素还是非常突出的。有了人工维修，失灵了的器件可以撤换，损毁了的部件或次系统可以得到修复。只要维修及时，包括两份次系统，即一份贮备的系统，下文中称这种系统为二模宽裕，其寿命应高出包括很多贮备而无人工维修的系统。

二模宽裕系统中，既然只有两份次系统，当二者协同操作而其输出相同时，人们可以认为输出是正确的，如果二者的输出有异，从这种差异本身无法区别何者正确，何者错误，必须对每一次系统赋予一种自我检查的机能，这就要求应用容错技术的其他方面，最常用的方法是对信息及/或对控制器输出的信号组合，加上一些宽裕，即附加一个或若干个奇偶位构成检误编码，下面对奇偶检误码子作一简介。

由于计算机所处理的或使用到的信息，不论是计算数据，机器指令或需予以控制调节的。物理化学参数都是已经用二进制数字编成码子的，不过对常规设计的机器来说，这些码子的每一位都代表着必不可省的信息内容，没有任何一个二进制符号是付余的，但象奇偶检查位就有些不同，它并不代表信息的任何内容，也不补充信息任何函义，其作用仅仅是为了统一一个码子里“1”的个数成为是奇数（或偶数）。

奇偶检查位是如何起帮助检误作用的呢？如果通过设置奇偶位的值，规定码子“1”的个数是奇数（或偶数），那么一旦出现一个具有偶数（或奇数）个“1”的码子，就知道有了错，当然具有奇数（或偶数）个“1”的码子不一定是不错的，它可能在两位上有错。

在无宽裕编码中，两个码子至少在一个二进位上有区别，我们说这种码子差距是1（$d=1$）。在包括奇偶检查位的码子中，两个不同码子至少在两个二进位上有区别，我们说这种码子的差距是2，（$d=2$）。如果在包括奇偶检查位的码子系统出现一个码子，它与本系统内另一码子只在一位上有所不同，这个码子必然是个错误的码子，错误经常是一位，但也可能是三、五……任何奇数位。

通过这个例子，我们认识到一个码子的奇偶检查位，是在各信息位以外，外加的位，或冗余的位。设置这样一个宽裕位，就可以帮助检查码子是否有错，严格的说帮助区别一个码子是否“合格”，合格码没有错，或者在两位、四位……上有错；不合格码在一、三……、位上有错。我们认为出错的概率本来是小的，不出错的概率最大，错一位的概率次之、错二位的概率更小……。所以一般认为奇偶位的设置可以帮助检出一位错。

奇偶检查位是个很好的宽裕技术例子，在码子上设置1位的宽裕，就能起帮助检查一位错的作用。这是一种最早被采用，一直到现在还广泛地被应用着的，有效而经济的宽裕技术。

在二模宽裕系统中，当两份次系统之一出错，即其输出信息为非合格码时，可以使它成为无效而用另一份的输出替代整个二模宽裕系统的输出[4]。这样就可以消除瞬间故障，从而延长其无故障运行时间，能延长到什么程度，可以应用下列数学模型来计算[5]。如图8

设单机的可靠率曲线为R，其平均无故障运行时间$MTBF$为τ，为了赋予自行检错的

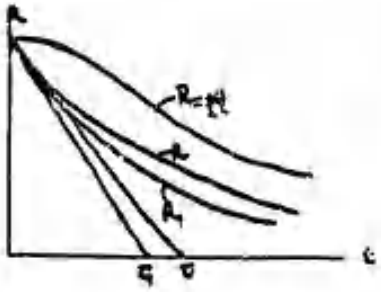


图 8 二模宽裕系统的可靠率曲线

机能，比方说在信息上加一个奇偶位，为此需要增添一些检错线路，其可靠率曲线有所降低，变为 R_1 其相应的平均无故障运行时间为 τ_1，当这样的两机协同操作互纠错误时，其可靠率为

$$R_{二模} = 1 - P_1^2$$

式中 $P_1 = 1 - R_1$，P_1 为不可靠率或失效率

所以

$$\begin{aligned} R_{二模} &= 1 - (1 - R_1)^2 \\ &= 2R_1 - R_1^2 \\ &= e^{-\frac{t}{\tau_1}}\left(2 - e^{-\frac{t}{\tau_1}}\right) \end{aligned}$$

平均无故障运行时间为

$$\begin{aligned} (MTBF)_{二模} &= \int_0^\infty \left(2e^{-\frac{t}{\tau_1}} - e^{-\frac{2t}{\tau_1}}\right)dt \\ &= -2\tau_1 e^{-\frac{t}{\tau_1}} + \frac{\tau_1}{2} e^{-\frac{2t}{\tau}} \Big|_0^\infty \\ &= 1.5\,\tau_1 \end{aligned}$$

由此可见二模宽裕的平均稳定时间比单机有所提高，优于三模宽裕。取 $R_{二模}$ 的导数得

$$R'_{二模} = -2\frac{1}{\tau_1}e^{-\frac{t}{\tau_1}} + \frac{2}{\tau_1}e^{-\frac{2t}{\tau_1}}$$

当 $t = 0$ 时

$$R'_{0二模} = -2\frac{1}{\tau_1} + \frac{2}{\tau_1} = 0$$

可见在 $t = 0$ 处，曲线的切线平行于 t 轴，这一点与三模宽裕相同。即对可靠率的要求愈高，二模宽裕系统，比起相应的单机系统来，其无故障运行时间延长得愈多。

修理不停机：以上关于无故障运行时间的计算，是没有考虑到人工修理的。二模宽裕系统的一个很重要的特点是无需停止整个系统的运行而对有故障的一机进行诊断、修复，修复后又可以参加协同操作。这样一来只要在修理过程中另一机不出故障就不致于脱节，系统的无故障运行时间可能是无限的了？

当然无限是不可能的，关于这一点可以作如下的计算：〔8〕

将故障分为两类：第一类是系统不整个停摆就可以排除的故障，这里面包括借助于协同操作而纠正的瞬间故障，也包括一机损坏但可以在另一机损毁以前，及时修好，不致使整个系统失灵的故障。第二类故障是导致系统整个停摆的故障。第一类故障占全部故障的绝大多数。我们把第二类故障与全部故障之比叫做双机的脱节率 D，在一般情况下 D 是个小数。

单机的可靠率 R_1 为：

$$R_1 = e^{-\frac{t}{\tau_1}}$$

式中的 τ_1 是单机的平均无故障运行时间。我们假定单机的修复过程也服从同样的规律：

$$R_2 = e^{-\frac{t}{\tau_2}}$$

式中 τ_2 是经过诊断后修复所需的时间，R_2 代表单机继续处于故障状态的概率。

若双机同时失灵，没有此机为他机进行诊断的机会，修理起来就需要更多的时间 τ_3，与此时间相应的修复规律为

$$R_3 = e^{-\frac{t}{\tau_3}}$$

在图 9 中①代表两机都完好的状态。命这一状态的可靠率，即两机都处于无故障状态的概率为 P_1；②代表双机系统中一机有故障，另一机完好的状态，命状态② 的可靠率，即双机系统处于状态②的概率为 P_2；③代表系统中两机同时失灵的状态，P_3 代表系统处于状态③的概率；④代表系统中两机先后失灵，但有条件对首先失灵的一机完成诊断的状态，P_4 代表双机系统处于状态④的概率。由状态④进行修理，其所需时间也是 τ_2。

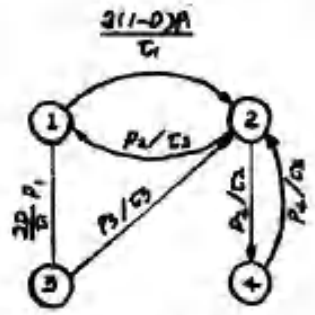


图 9 可修理双机协同系统的可靠率模型

如果不考虑修复的效果，双机系统处于状态①的概率将是：

$$P_1 = R_1^2 = e^{-\frac{2t}{\tau_1}}$$

但因为失灵的那一机可以修复，P_1就不如此简单，不好直接写出，倒是 P_1 的时变率 $\frac{dP_1}{dt}$ 或 P_1' 容易写出，P_1' 是双机系统离开状态①的速度。图 9 中箭头表示变化的方向，箭头上所书写的是变化的速度。显然，对于任一状态而言，到达的箭头所表的速度是正数，离去的箭头表示的速度是负数，因此：

$$P_1' = -\frac{2}{\tau_1}e^{\frac{-2t}{\tau_1}} + \frac{1}{\tau_2}e^{\frac{-t}{\tau_2}}$$

$$= -\frac{2P_1}{\tau_1} + \frac{P_2}{\tau_2}$$

但 $2P_1/\tau_1$ 由两部分组成：一部分 $2DP_1/\tau_1$，导致状态③相应于直接引起系统停摆的故障；第二部分 $2(1-D)P_1/\tau_1$，导致状态②，相应于系统在运行中即能修复的故障。

另外，③，④都代表双机失灵状态，但③与④有所区别：两机同时失灵就进入状③态，先后失灵就进入状态④。

现在对 τ_1，τ_2，τ_3，赋予一些假定数值：假定构成双机系统的两台机器是相同的两台机器，每台用 1000 个双与非门组件做成，为了简化计算，排除整机制作工艺水平的影响，把组件失效率定得高一些如 10^{-5}/小时，这样，单机的平均无故障运行时间 τ_1 约

为 $10^{6-3}=1000$ 小时。

对于一台整机，即使是千数片子的小机器，诊断程序（停机诊断程序而不是操练程序）一般是比较庞大的。我们假定在它的硬件设计中，已考虑到可以由软件将它划分为较小的分割以利诊断，可以把故障所在明确到少数个点，在这种情形下，修复时间 τ_2 不会太长，例如 $\tau_2=1$ 小时，至于修复一台未经诊断的机器，时间可能要长些，这关系到维修人员的技术熟练程度。假定 $\tau_3=8$ 小时。下面写出图 9 中各状态的时变率：

$$P_1'=\frac{-2}{\tau_1}P_1+\frac{1}{\tau_2}P_2=0$$

$$P_2'=\frac{2(1-D)}{\tau_1}-\frac{1}{\tau_2}P_2-\frac{1}{\tau_1}P_2+\frac{1}{\tau_3}P_3+\frac{1}{\tau_2}P_4$$

$$P_3'=\frac{2D}{\tau_1}P_1-\frac{1}{\tau_3}P_3$$

$$P_4'=\frac{1}{\tau_1}P_2-\frac{1}{\tau_2}P^4=0$$

当 4 个状态之间平衡建立之后，各状态的时变率都应当是零：

$$\begin{aligned}
&-\frac{2}{\tau_1}P_1+\frac{1}{\tau_2}P_2=0\\
&\frac{2(1-D)}{\tau_1}P_1-\left(\frac{1}{\tau_1}+\frac{1}{\tau_2}\right)P_2+\frac{1}{\tau_3}P_3+\frac{1}{\tau_2}P_4=0\\
&\frac{2D}{\tau_1}P_1\qquad\qquad-\frac{1}{\tau_3}P_3=0\\
&\qquad\frac{1}{\tau_2}P_2\qquad\qquad-\frac{1}{\tau_2}P^4=0
\end{aligned}$$

这是一组联立齐次方程，易于看到各项的系数之和是 0，因此 P_1，P_2，P_3，P_4 不是相互独立的，可以用其一表达其他，例如：

$$\left.\begin{aligned}
P_2&=\frac{2\tau_2}{\tau_1}P_1\\
P_3&=\frac{2D\tau_3}{\tau_1}P_1\\
P_4&=\frac{\tau_2}{\tau_1}P_2=\frac{2\tau_2^2}{\tau_1^2}P_1
\end{aligned}\right\}\qquad(A)$$

在一年的时间中，双机系统处于状态①，②的时间为运行时间 T，消磨在状态③，④的时间为系统的停摆时间 T_o，若以年为时间单位，则 $T_o+T=1$。基于此停摆时间在全部时间中所占比值为：

$$\frac{T_o}{T+T_o}=T_o=\frac{P_3+P_4}{P_1+P_2+P_3+P_4}$$

由于(A)的关系：

$$T_o=\frac{2}{\tau_1}\left(D\tau_3-\frac{\tau_2^2}{\tau_1}\right)\Big/1+\frac{2\tau_2}{\tau_1}\left(D\tau_3+\frac{\tau_2^2}{\tau_1}\right)$$

根据对 τ_1，τ_2，τ_3 所作的假定：

$$T_o \cong \frac{2\tau_3 D}{\tau_1} \qquad (B)$$

由 (B) 可以看到关修复影响大的是由状态③修复，而不是由状态二修复。从 T_o 可以算出每年修理次数，即停摆次数 N：

$$N = \frac{T_o}{\tau_3} = \frac{2D}{\tau_1} \qquad (C)$$

每年运行时间为：

$$T = 1 - T_o = 1 - \frac{2\tau_3}{\tau_1} D$$

最后，平均无故障运行时间 $MTBF$ 应为每年运行时间除以每年停机次数：

$$MTBF = \frac{T}{N} = \left(1 - \frac{2\tau_3}{\tau_1} D\right) \frac{\tau_1}{2D} \qquad (D)$$

即

$$MTBF = \frac{\tau_1}{2D} - \tau_3 \approx \frac{\tau_1}{2D} = \frac{1000}{2D} \text{小时} \qquad (E)$$

对各种不同的 D 值，$MTBF$ 的数值如下：

D	$MTBF$（小时）	$MTBF$（年）
1.0	500	0.057
0.5	1,000	0.114
0.1	5,000	0.57
0.01	50,000	5.7

部件阶的二模宽裕〔4〕

上面描述的二模宽裕是属于整机阶的宽裕。它的优点在于具有较高的灵活性，使用单位购置两台相同的具有奇偶检错的机器，加上些接口线路，实现协同操作就成了。

如果在机器的设计制造过程中，采用两套具有自检错机能的部件就形成一台双部件机器，拥有部件阶的二模宽裕。

部件阶的二模宽裕比起整机阶的二模宽裕有下列几种优点：

(1) 自诊断机能，部件阶的二模宽裕，实际上就是部件阶的协同操作，可以在部件一阶相互纠错。如果在这个基础上，增设检错的记录线路，为了经济可以采用巡回记录的方式，在某一部件，在一定时间内出错次数超过一定的限额或其出错时间超过一定限值，就可以认为该部件有了固定性故障而给出警报信号，这就在机器的运行中完成了自诊断工作的一大部分，而且是最艰巨的部分。另外设立一个诊断站，将有固定性故障的部件，包括一个或少数个扦件，一个一个地从机架上拔下来，扦到诊断站的扦座上，对它或依次对它们进行诊断。可以把诊断站看成是机器的一个外部设备。诊断工作可以通过程序中断而进行，不妨碍机器算题工作的照常进行，这样诊断工作就被限制在扦件诊断的范围，且所需时间较之对全机的诊断可以大大缩短，从而大大加速维修过程。再则，对扦件诊断所需诊断程序，比对全机的诊断程序，要简单得多，短得多，占用的存贮量要小得多，不致对内存造成负担。

(2) 延长平均无故障运行时间，平均无故障运行时间的计算方法，大致与双机系统相同，不过值得注意的是公式：

$$MTBF \approx \tau_1/2D$$

其中 D 的数值大不相同了。D 所代表的是脱节率，譬如说，双机协同系统中每一台机器，由 20 个部件构成，总共 40 个部件。

在部件协同的双部件单机中也包括这样的 40 个部件。但在双机系统中甲机的第 i 个部件，$i=1\sim20$，失灵，甲机就告失灵；与此同时若乙机的第 j 个部件，$j=1\sim20$ 失灵，乙机也告失灵，从而整个系统就停摆。

对于双部件单机来说，在第 i 个部件失灵后，在剩下的 39 个部件中必需是与第 i 个配对的那一个也告失灵，整机才算失灵。一个最极端的说法是：全机的 40 个部件中，失灵部件多至 20 个，只要没有两个配对的部件同时失灵，机器就能运行，这就是说双部件单机比双机系统的 D 要小 N 倍，平均无故障运行时间要高 N 倍，N 是配对部件的对数。

要付出什么代价才能获得那么大的好处呢？代价是保证协同操作的交叉接口线路增多了，这些线路可能要占整个宽裕量的百分之十几。

2) 通用计算机类型的系统，它的主要指标是单位时间内的信息流通量，因为待解决的问题有时非常庞大，所以也要求有尽量长的无故障运行时间，但计算工作基本上是分批量进行的，可以分段落的，因此停机维修是允许的。另一方面要迅速恢复工作，以提高设备的使用效率，即提高其“可用性”。

这类机器多数是比较庞大的系统，要在硬件上采用很多的宽裕技术，从经济的角度考虑往往不允许，目前我国集成电路的集成度还不高，尤其如此。另一方面机器的外围设备较多，而外围设备的故障率总是比较高的。在这种条件下，在主机上采用很多的宽裕技术，也是不相称的。当然对外围设备也可以考虑采用一些容错技术[6]。总之对这种机器可以说信息通量与高度的可用性是两个同等重要的指标。

提高可用性的手段主要在于迅速修复，减少停机时间，而这又靠迅速明确故障在那里，即要求迅速进行故障诊断，这个问题可以说从理论上已经得到解决。著名的故障定位方法有 D 算法，布尔差分等[7]，[8]，[9]。

在应用这些故障诊断方法时，都遇到一个实际上的困难，即诊断程序包括故障仿真、测试序列产生等环节比较庞大，要占内存的过大面积。

对于故障诊断最近呈现两种趋势：一是在计算机系统的硬件设计中考虑到故障诊断的问题，向它提供方便的条件；二是由于集成电路集成度的提高，对整机诊断的要求有所降低，只要求故障定位到组件就可以，而无须定位到逻辑的一级。

由于集成度的提高，在组件的生产测试过程中，必须考虑对组件内部线路进行诊断，来摸清组件的那些部分易于出问题，对这些薄弱部分的改进可以导致更高的产品合格率。

便利组件内部诊断的主要措施是 1) 在组件内部的器件之间建立两种可以互相切换的连接关系：一种是正常运行时的关系，另一种是诊断故障时的关系，例如将组件中的记忆原件连接成移位寄存器，只需增添较少的线路就解决记忆原件的诊断问题，总之

为了建立两种连接并互相切换，是要增加一些线路与测试点的，据报导，所增加数量，在某种情况下大致不超过8%。2) 保留组件内部的连接，将若干组件组成一个次系统。每个次系统备有它自己的测试向量源和控制源，控制着诊断测试与正常工作二者之间的分时操作。每个组件具有它自己的测试向量检测逻辑，它产生一个码子，代表着本组件各输出端及关键性附加测试点对一个测试序列的反应，将这个码子和事先计算出来的正确结果相比较，就得到组件的通过/失灵信号。

根据 ILLIACN 机开件测试工作中所获得的经验与 Aughes 公司的研究结果，只要被测试组件中门的串连级数＜8，一个任意的即近乎随机的测试向量序列能够几乎比一个有针对性的测试序列同样完善地操练逻辑线路[10],[11]，既然测试向量序列可以是不带针对性的，那就不必要用只读存贮器(ROM)来存贮这些计算出来的模式，而可以用较少的器材，做成带适当反馈通路的移位寄存器，作为某种质多项式的循环编码器，来生成近乎随机的向量序列。

以上方法不仅适用于集成电路组件的测试诊断，也适用于开件的诊断，原则上也适用于整机系统的诊断。

除了保证一定的可用性以外，在通用计算机系统中是不是也可以采用一些容错技术以提高它的无故障运行时间呢？答案当然是肯定的。

当奇偶检查器检出错误后，计算机可以自动的采取适当措施。例如使当前指令重复执行若干次，如果故障是瞬间性的，经过一段不长的时间，即自行消除，那么奇偶校验结合"指令复执"就有可能挺过一段出错的瞬间而使机器恢复正常运行，这就延长了无故障运行时间，提高机器的可靠率。如果故障不是属于瞬间性，而是属于永久性的或固定性的，那么奇偶检验器所检出的出错信号，告诉程序员机器在执行某指令时出错的，应当采取相应措施进行挽救，如果可能的话，若不然则应进行修理重复开始计算等，这样就能使计算机的工作不致一错就错到底，这也是一种提高可靠性的形式。

关于"指令复执"有以下几点必须予以注意：

不是每一条指令或在其执行过程的任何阶段都是可以重复执行的，必须在本指令开始时机器的"现场"未变的情况下才能进行"复执"。所谓"现场"包括与本指令有关的各个寄存器，程序步进计数器，以及其他有关计数性单元的状态。否则在每一指令的开始就要在内存中保留各有关寄存器的状态，这是很费时间的。

对于所谓瞬间故障的持久性往往考虑不足，从某些既得的宝贵经验看来，复执8次或16次无效就认为出现了固定性故障是欠妥的，可能需要重复几十到几百毫秒。

在一条指令重复多次也消除不了瞬间故障情况下，就有理由怀疑错误的产生可能不在本指令的执行过程中，而应当向前追溯，这样就导引到程序卷回的概念。

程序卷回不是某一条指令的重复执行，而是一小段程序的重复执行，为了实现程序卷回，亦必须保护其现场。此处现场的范围就更广一些。具体一点说这保护现场的办法可以如下实现：(1) 将程序分成一小段一小段的，卷回起来也就卷回一小段。(2) 在第 n 段之末将当时各个寄存器指令计数器及其他计数器等的内容移入内存，将内存中为第 n 段所更改的单元，如中间结果，计数单元内容等亦在内存中另行存贮，保留起来，这叫作存档。(3) 如果在第 $n+1$ 段中不出问题，则将第 $n+1$ 段存档，注销第 n

段的存档。（4） 如果在第 $n+1$ 段中出了错，就把第 n 段的档，回送到机器的相应部分，然后从第 $n+1$ 段的起点开始重复执行第 $n+1$ 段程序，这就是程序卷回。

通过一个奇偶位的检查结合指令复执或程序卷回，可能获得可靠率提高的估算。

设原系统没有奇偶检查，其可靠率函数为

$$R = e^{-\frac{t}{\tau}}$$

添加了奇偶位之后，τ 值应有所降低，但由于所加器件较少，仅约 5～10%，奇偶位加复执所增线路的可靠率应为

$$R_f = e^{-\frac{t}{10\tau}}$$

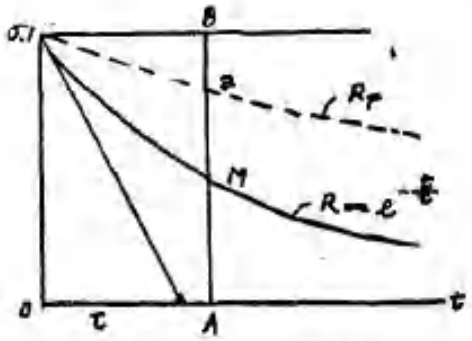


图 10 指令复机或程序卷回对可靠率的提高

另一方面，根据个别单位的经验，指令复执所能排除的瞬间故障约占全部瞬间故障的 70～80%，这个比值、有待在今后的经验予以进一步明确，另外瞬间故障约占全部故障的 70～80% 这一数值也属于某些单位的经验之谈，但未经统计，只有作为参考的价值。

设 $OA=t$，$AM=R$，$MB=1-R=P$

MB 代表全部故障，瞬间故障率 = $(0.7\sim0.8)\times P$ 瞬间故障中可以“复执”排除的部分 $=0.8\times0.7\ P=0.56P$

复执所需线路本身的可靠率 $R_f=e^{-\frac{t}{10\tau}}$

取 $MQ=0.56\times MB\times e^{-\frac{t}{10\tau}}$ MQ 代表着指令复执技术由可靠率上所获得提高，对各不同 t 值，Q 点的轨迹的方程应为

$$\begin{aligned}R_F &= AM+MQ=e^{-\frac{t}{\tau}}+0.56\,P\,e^{-\frac{t}{10\tau}}\\ &=e^{-\frac{t}{\tau}}+0.56\left(1-e^{-\frac{t}{\tau}}\right)e^{-\frac{t}{10\tau}}\\ &=e^{-\frac{t}{\tau}}+0.56\ e^{-\frac{t}{10\tau}}-0.56e^{-\frac{11}{10}\ \frac{t}{\tau}}\end{aligned}$$

有了复执或卷回平均稳定时间将为：

$$(MTBF)_{复执}=\int_0^\infty R_F\,dt=\left[\tau+0.56\times10\tau-0.56\frac{10}{11}\tau\right]\cong6.1\tau$$

从 τ 值被提高 6 倍及曲线(R_F)的高度来看，应给予指令复执（或程序卷回）以极高的平价，尤其因为所需增添的设备量有限。应当注意到指令复执对固定性故障不能起排除作用，而只能起检出作用。

容错技术应用于存贮器，在计算机中存贮器发生故障的概率显著地超过运算控制的

故障概率，因此在存贮器中设置宽裕以检出或自动纠正错误，在一定条件下，可能是比较恰当的，在自动纠错的方案中，当一个具有一位错的字从存贮器读出时，在存贮器系统内部，立即将它纠正，然后传送到处理器，因此对处理器的操作无影响。我国正在研制中的系列机已采用了这种技术，即便是小型计算机，如果它需要比较长期地无故障运行，也应考虑对存贮器进行自动纠错，因此对存贮器自动纠错方案的工作原理，应付出的代价与可靠率的提高进行分析，是在设计计算机整体方案时值得考虑的。[12]

工作原理：自动纠错系统不过是奇偶检查的推广：将各信息位分组，对每组添加一个奇偶位。该各奇偶位在写入前自动产生，写入时列于信息字的尾部，例如共有信息位16个，a_0，a_2，…a_{15}将各信息位分成5组，并配以5个奇偶位如下

奇偶位 \ 信息位	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}
y_0	1	1	0	0	1	1	1	0	0	0	1	1	1	0	0	0
y_1	1	1	0	0	1	0	0	1	1	0	1	0	0	1	1	0
y_2	1	0	1	0	0	1	0	1	0	1	0	1	0	1	0	1
y_3	0	1	0	1	0	0	1	0	1	1	0	0	1	0	1	1
y_4	0	0	1	1	0	0	0	0	1	1	1	1	1	1	1	1

即命

$$y_0 = a_0 + a_1 + a_4 + a_5 + a_6 + a_{10} + a_{11} + a_{12} \; Mod\, 2$$
$$y_1 = a_0 + a_1 + a_4 + a_7 + a_8 + a_{10} + a_{13} + a_{14} \quad 〃$$
$$y_2 = a_0 + a_2 + a_5 + a_7 + a_9 + a_{11} + a_{13} + a_{15} \quad 〃$$
$$y_3 = a_1 + a_3 + a_6 + a_8 + a_9 + a_{12} + a_{14} + a_{15} \quad 〃$$
$$y_4 = a_2 + a_3 + a_{10} + a_{11} + a_{12} + a_{13} + a_{14} + a_{15} \quad 〃$$

编码时信息16位来自累加器AC，编码操作为：

$y_0 \leftarrow \Sigma mod\ 2\ AC(0,1,4,5,6,10,11,12)$
$y_1 \leftarrow \Sigma mod\ 2\ AC(0,1,4,7,8,10,13,14)$
$y_2 \leftarrow \Sigma mod\ 2\ AC(0,2,5,7,9,11,13,15)$
$y_3 \leftarrow \Sigma mod\ 2\ AC(1,3,6,8,9,12,14,15)$
$y_4 \leftarrow \Sigma mod\ 2\ AC(2,3,10,11,12,13,14,15)$

奇偶位y_0…y_4附在信息位a_0…a_{15}之后并写入存贮器，读出时共有21位，读到MJ后进行译码：

$z_0 \leftarrow \Sigma mod\ 2\ MD(0,1,4,5,6,10,11,12) \oplus y_0$
$z_1 \leftarrow \Sigma mod\ 2\ MD(0,1,4,7,8,10,13,14) \oplus y_1$
$z_2 \leftarrow \Sigma mod\ 2\ MD(0,2,5,7,9,11,13,15) \oplus y_2$
$z_3 \leftarrow \Sigma mod\ 2\ MD(1,3,6,8,9,12,14,15) \oplus y_3$
$z_4 \leftarrow \Sigma mod\ 2\ MD(2,3,10,11,12,13,14,15) \oplus y_4$

纠错时$(z_0 \cdots z_4)$构成指错字(000000)代表合格码$z_0 \sim z_4$的位中若只有z_t一位为"1"，

表明读出的奇偶位 y_i 就有错而信息无错，必须纠正，由前列公式，导出纠错线路如下：

$$AC_0 \leftarrow z_0 \cdot z_1 \cdot z_2 \oplus MD_0$$
$$AC_1 \leftarrow z_0 \cdot z_1 \cdot z_3 \oplus MD_1$$
$$AC_2 \leftarrow z_2 \cdot z_4 \oplus MD_2$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$AC_{16} \leftarrow z_2 \cdot z_3 \cdot z_4 \oplus MD_{16}$$

代价：增添多数个奇偶位增加了字长，增加功耗增加线路包括编码译码线路示错自动归零寄存器等，另外自动纠错影响存贮器的读写周期：写入时由半加器组成的编码器需要对各组中的个数进行按位加，从而产生各分组奇偶位然后才写入；读出时要对各奇偶位形成的指误字进行检查，必要时对错位进行纠正，然后才能将数据送向处理器，这将使访问时间，读写周期增加 30 到 150 ns。

好处：一位错的纠正能够有效地提高可靠率。为了精细地估算提高数字，有必要区别“硬错”与“软错”，字间的错或位间的错，另外还要明确“故障率”的函义，放弃笼统的、不变的“平均无故障”运行时间($MTBF$)的概念，同时还要考虑到自动纠正一个固定性错误时采取什么措施。

有些错是不能纠正的，例如输入/输出有错，在未产生纠错奇偶码以前信息就有错，或者在纠正错误之后又出错。又如选字译码器的故障导致读错了字或将一个字写入错误的地址，象这样一些字与字间的错也属于总的故障率组成部分。

至于位间的错，也就是奇偶位能够纠正的错，要考虑它能影响到的字数，如果存贮体分成小块，一个单个的错只会影响到一个分体内的字，若各分体共用一个数码寄存器 MD，MD 的一位故障将影响到全部字。

只要存贮器的其他部分无故障，这些错都能得到纠正，但如果已经有了影响到很多个字的固定故障，当第二个故障发生时，在同一个字中发生两个故障的概率显然会上升。

故障率分析：假定一个半导体存贮器具有 128 k 字，每字 16 位，由 4k 位的片子组成，共 512 片。

假定所有故障都是硬错，每个错影响到系统中全部字的 5 %，器件的故障率为每 1000 小时。 0.01%

如果没有纠错装置，在 1000 小时内，总的故障率将为 $512 \times 0.01\% = 5.12\%$，也就是说平均稳时间为 $\frac{100}{5.12} \times 1000 = 19.53$ 小时。

采用单故障纠正时，字长变为 21 位，在第一个 1000 小时内出单个错的概率上升到 $21/16 \times 5.12\% = 6.72\%$，但这些错将被自动纠正，处理器将感觉不到。只有在同一个字中出现第二个错时计算机才会见到错。

既然典型的固定错影响到地址场的 5%，这 5% 的字成为易于产生第二固定性故障的地区，因为第二故障也有它 5% 的影响范围。这样，先后产生的两个固定性故障重合在一个字上的概率将是 $2 \times 5\% = 10\%$。

第二个故障产生的概率，因为它只能产生在剩下的 20 位，应当是 $20/16\times 5.12=6.4\%$。综合以上三点，在运行的第一个千小时内，两个独立的固定性故障重合于一个字的概率应当是

$$6.72\%\times 6.4\%\times 10\%=0.043\%$$

它相当于

$$MTBF=100/0.043\times 1000=2,420,000 \text{ 小时}$$

比没有纠错系统的 *MTBF* 19,530 大两个数量级。

以上分析，对于固定性故障的影响范围，对于它的分布情况是很敏感的。显然，如果第一故障影响到存贮器的每个字，第二个故障，即便是偶然性的故障也将形成在一个字内的双重故障即不能纠正的故障，而且在出现了第一个固定性的故障之后，新的故障率将比没有纠错的存贮器的更高，因为位数是 20 而不是 16。

以上分析只适用于最初的 1000 小时，即不存在一个固定性的，被自动纠正的故障的一段时间，随着运行时间的延长，已发生一个故障的概率按指数规率增长，既然产第二个故障的概率是常数，产生双重故障（即不能纠正的故障）的概率也将按指数规率而上升。在上述的例中，最初 1000 小时的故障率为 6.72 %，在运行 5000 小时之后，一个“透明”的，即被自动纠正的固定性故障已经发生的概率大致是

$$1-(1-0.0672)^5=0.294=29.4\%$$

在第六个千小时内，出现双重故障的概率将是

$$29.4\%\times 6.4\%\times 10\%=0.188\%$$

失效将上升 $29.4/6.72\approx 4.4$ 倍，即 *MTBF* 将缩短 4.4 倍。

实际上，在两个或多数个不同的字中产生单个的固定故障也有一定的概率。尽管这些故障能够得到自动纠正，它们使发生双重故障的概率上升。

自动纠错的主要优点在于纠正瞬间单故障而不留痕迹，但如果存贮器中已经存在着较多的固定单故障，一旦与瞬间单故障重合，即变为不能纠正的双重故障。为此，要通过定期修理来消除固定单故障，也可以从以上分析出发来决定设计存贮器时要不要采用自动纠错技术。

参 考 资 料

〔1〕 Triple modular redundancy networks, Reliability evalation, T—C 74, July,

〔2〕 A highly Effirient redundancy scheme: Selfpurging redundancy, T—C 76, June.

〔3〕 A reliability model for gracefully degrading and standby-sparing septems• T—C 75, May.

〔4〕 RC—1 型计算机的设计方案，哈工大 43 专业。

〔5〕 二模宽裕系统的可靠率模型，同上。

〔6〕 A damage-and fault-tolerant Jnput/output network, T—C 75, May.
〔7〕 Faut diagnosis of digital systems—H. Y. Chang, E. Mauning & G. Metze, Wiley Interscience, 1970.
〔8〕 Algonthm for detection of faults in logic circuits T—C 71. Nov.
〔9〕 Analyzing errors with the Boolean Difference T—C 68, July.
〔10〕 Automatic test generation system for Illiac IV logic boards, T—C 72, Sept.
〔11〕 Adranced Arionics fault isolation system T—C 75, May.
〔12〕 Auaomatic error correction in memory systems Comp. Design. 1976, May.