

电脑与法律

从几件事看我国软件保护研究近况... 成 海 (2)
专论

如何选购一台好的电脑显示器(上)
 陈文斌 (3)

电脑应用

条形统计的直接打印..... 王有翥 (7)
 四端网络幅频特性的自动测试系统
 李冰清 (10)

软件介绍

通用动态表格处理系统 SJ-TAB ... 沈 军 (12)

网络与通讯

高级 UNIX 连网技术讲座 第二讲 TCP/IP
 (二) 冯家宁 (15)

书讯

电子工业出版社广州科技公司邮购书讯 ... (19)

中文信息处理

在真正的 MS DOS5.0 或 DR DOS6.0 下运行王
 码 5.0、金山 5.0、2.13H 等汉字系统
 宋运康 (20)

使用与维修

警惕 CHKDSK 报告的簇丢失及交叉错误
 徐永久 (23)
 中华学习机维修一例 王 曦 (23)
 DOS 混装软故障初识..... 彭 禾 (24)
 一种查找 LQ-1600K 打印机故障的简捷方法
 梁 军 (25)
 如何快速更换 LQ-1600K 打印针
 王英伟 (26)
 CR3240 打印机故障析修两例 杨轩举 (26)

IDEA

如何将四通电脑打字机上编辑的文件在微机上使用
 皮可慰 (27)

编读往来 (28)

大学生之页

控制程序执行次数的两种方法 周国均 (29)

SOURCE4.56 安装号之拆解 朱 彤 (30)

中学天地

中华机固化 DOS 研究 王建民 (31)

电脑辅助教学

功率谱估计的 C 语言实现 张红庆 (33)

用 dBASE III 评估学生成绩 魏明逊 (37)

竞赛与考试

一九九三年广东青少年信息学(计算机)竞赛初
 赛试题分析及其参考答案 邬家炜 (39)

游戏乐园

煮酒论英雄——大型游戏“三国演义”全略
 卫 易 (42)

病毒防治

第二代病毒的产生及其对策 何善华等 (44)

单片机与单板机

单片机与模糊控制讲座 第三讲 模糊控制
 理论 余永权 (48)

电脑用户

数据库卡片式查询通用程序 莫胜强 (51)
 Turbo C 在非中文系统下显示和放大汉字的实现
 邱素刚 (53)
 利用不可逆函数产生密码 罗永强 (55)
 用键盘中断解决“死机”问题 张 研 (57)

广告索引 (6)

从几件事看我国软件保护研究近况

本刊记者 成海

我国软件保护条例施行虽不到两年,软件保护研究的进展却比较明显,处于“加速度”状态,已取得了一定的成果。这些研究,有的偏重于法律,有的偏重于软件技术,有些则是两者的结合。其内容涉及国内外的理论、实践、案例,论者们或摆明观点,或分析论证,或提出方法,或总结经验……总之,很有生气。

记者现在仅就近半年多来的几个较有意义的事件,说明这一状况。

第一、《计算机软件著作权保护工作手册》一书的出版。

该书由中国软件登记中心和中国计算机报社主编,电子工业出版社1993年2月出版,计三十五万字。机电部副部长曾培炎为该书作序。全书分为三个部分,第一部分汇集了国家和有关部门业已颁布实施的关于著作权和计算机软件保护及登记的法律、条例、法规。第二部分是计算机软件著作权登记指南,包括登记法的内容、要求,具体的登记办法、手续,收费标准及登记表格样式等等。第三部分是国外计算机著作权纠纷案例选编,共选了美国、日本、澳大利亚、荷兰、挪威、以色列等国80年代以来的较典型的案件共31例。影响较大的如:Apple公司诉Franklin公司(关于软件版权的基本对象),NEC诉Intel公司(微程序的版权问题),Whelan公司诉Jaslow(S.S.O原则),CA公司诉Altai公司(对S.S.O准则的另一观点),日本富士通与美国IBM公司的软件纠纷(操作系统的接口规格保护问题,由仲裁解决),Sega公司诉Accolada公司(逆向工程侵权问题),Lotus公司诉Paperback公司(用户界面的版权)等著名案例,都收集在这里面。其它还有牵涉到雇员行为,屏幕设计,“独创性”判断,“外观与感受”,合同关系等方面的案例。在案例前后,还夹有编译者的议论。

该书既可以看作是我国官方为软件保护所做的工作的一个缩影,又是国外软件保护情况的资料精选,确是编者们研究工作的一个硕果。

第二、《计算机世界》周刊于1992年最后一期(51期)刊出关于软件保护的专题版。

该版包括了6篇文章:高技术企业面临的知识产权管理工作(应明),论计算机软件著作权人的权利(寿步),技术转让费的确定及支付方式(马志毅),软件的著作权保护对软件用户的影响(李勇),计算机软件出版中的有关法律问题(李维),高科技与知识产权(叶燕)。

这组文章从理论到方法对知识产权和软件的法律保护进行了讨论,每篇文章都有较大的篇幅,因而对提出的论

题都讨论得较深刻,有些还针对具体的问题逐一作出回答(如李维文),对软件保护研究有很大的参考价值。《中国计算机报》在这段时间内,也刊登了软件保护方面的文章。

第三、《著作权》杂志(季刊)中软件保护方面文章占有相当比重。

在93年第一期中,有“美国对软件版权纠纷判处的新发展”(应明),从美国计算机程序著作权判例法的发展探索程序著作权保护的范围(金渝林)。在93年的第二期中,有“论同一作品的两种不同表现形式”(金渝林)。在该期刊登的全国人大教科文卫委员会的关于《中华人民共和国著作权法实施情况检查报告》中,也专门谈到计算机软件保护的问题。

第四、一些计算机专业性很强的杂志开始刊登软件保护的文章。

《计算机科学》去年10月(第5期),刊登了“软件相似性判别准则之讨论”(王桂海)一文。计算机专业学术杂志刊登软件保护研究文章,过去如果不是没有的话,也属很罕见的。这篇文章的发表,说明软件保护已在法律的基础上开展了专门性的方法与技术研究。据了解,计算机学会会刊之一的《计算机应用与软件》也将刊登这类性质的文章。

第五、'93中国版研究会学术年会上,计算机软件保护的文章占有相当的份量。此次会议共有应征论文113篇(软件占11篇),推荐在会上宣读的论文26篇(软件方面4篇)。这批论文内容有“软件反向工程的合理性”(邹竹)“软件著作权受侵害后的赔偿”(应明),“计算机病毒与计算机软件保护”(李议,张建申),以及牵涉到软件表达形式,软件著作权概念的构成条件,软件产权保护与合理使用,软件著作权客体分析,软件合同等。这次学术年会的与会者普遍认为我国软件保护研究近期有了较快的发展。

第六、若本刊开设《电脑与法律》专栏,也忝列其后,则算是又一件有意义的事。

本刊过去曾陆续刊登了一些关于软件保护的社论、评论、专题论文以及通俗性介绍。为适应形势,从今年起开辟了《电脑与法律》专栏,其目的在于推动我国软件保护研究(尤其是基于计算机科学这方面的研究)的发展与交流。半年来对有关程序侵权,软件保护范围动态等方面,作了分析和报导,已有些读者(包括青少年读者)来信表示对这一栏目感到兴趣。

以上这几个事例,可以在一定程度上,从某几个侧面反映我国软件保护研究的成果及其活跃、深入的状况。这确是令人振奋的。

如何选购一台好的电脑显示器(上)

陈文斌

在微型电脑系统逐渐走入办公室，学校，家庭，个人的潮流中，任何人都会觉得它协助我们处理公文，分析，储存资料以及管理个人信息事务时的能力和速度而极欲拥有一套微型电脑系统。而在各式各样组成微型电脑系统的配套中，我们与电脑主机沟通的一个重要组件，就是电脑显示器。

十余年前，当第一部苹果牌微型电脑自车库中发展成功后，人们就注意到它可能掀起信息界的浪潮。而电子工程师们一方面不断地改良它的速度与容量，一方面注意到它功能的扩张性，而不断地加大它的应用范围。

负责双向沟通的显示器也不例外，由单色（绿色，琥珀色，纸白色..）演变到彩色，再注意到彩色显示器的分辨率（RESOLUTION）而改良显像管，由缩小其素图（PIXEL，构成图像的元素，即荧光点，每组像素中包含有红，绿，蓝三种荧光点。）的点阵（DOT PITCH，像素中相隔两绿点间距离，如图2红，蓝亦同），增加它的扫描条数以达高解析能力的要求。一切努力，不外乎提高彩色显示器的画面质量，与兼容性（COMPATIBILITY，即电脑接口卡相通的能力）因而使电脑选购者在这五花八门的产品中，感到困惑。

技术规格

彩色显像管		尺寸（对角线）	14 英寸，90 度偏转						
		荧光屏	淡墨色彩消光，不反射映管（化学腐蚀或砂质涂膜）						
		点距	TKM-1438AS / TKM-1438AR:0.28mm TKM-1438AM:0.39mm						
		荧光粉	P22，中短余辉						
		透光率	中心位置约 57%						
同步信号		波形	TTL 正负极性						
		水平频率	31.5 / 35~38kHz（千赫）						
		垂直频率	50~90Hz（赫）						
视频信号		波形	R.G.B.正极性，模拟式						
		波幅	0~0.7VPP						
		频带宽度	最大 45 兆赫（1024×768Mode）						
分辨率	Mode	1	2	3	4	5	6	7	
	水平像素× 垂直条数	640×350	640×400	640×480	1024×768	800×600	640×480	800×600	
	水平频率	31.47kHz	31.47kHz	31.47kHz	35.52kHz	35.16kHz	37.8kHz	37.88kHz	
	垂直频率	70.08Hz	70.08Hz	59.95Hz	86.96Hz	56.25Hz	72Hz	60.3Hz	
显像颜色					无色				
显像区域					255mm×190mm（1024×768Mode）				
兼容性					IBM VGA 8514 / A SUPER VGA，VESA standards				
输入电源					90~267 V AC，47~63Hz 全自动交换式电源系统				
功率消耗					<85 瓦				
信号端子					15-Pin D-Shell Connector				
使用环境温度					0~35℃ 室内温度（ambient）				
使用环境湿度					10~90%无凝结水滴（Non-Condensing）				
可控制部份		前控板	电源（Power），亮度（Brightness），对比度（Contrast），水平相位（H-Phase），垂直高度（V-Height），垂直中心点（V-Center）						
		后控板	水平相位-2（H-Phase 2），水平相位-3（H-Phase 3），35 千赫 / 38 千赫开关（35kHz / 38kHz Switch）						
整机尺寸					367mm×360mm×390mm（Without base）				
重量					TKM-1438A: 12.5kg				
附件					转盘，电源线				

选好电脑系统后，如何选择一部帮助你与电脑连接沟通的显示器，这里，笔者愿意向您提供一个简易地辨识显示器功能、性能的方法：

首先请你在决定好你选择的电脑系统，（如 IBM 的 PS/2 或苹果牌的麦金塔—目前世界最普遍的二代微型电脑系统）及接口卡（INTERFACE CARD）后再决定你最重要的显示器。此时请先翻阅你所想购买的显示器目录（CATALOG），一般的规格资料如下表，我们来解释一下表中各项的意义（以 38KSVGA 为例）。

(1) 彩色显像管：

(A) 14 英寸，90 度偏转：它告诉你这部显示器荧光屏的对角线的长度是 14 英寸（实际上荧光粉涂层面积对角线只有 33.54 公分），扫描偏转角是 90° 。

(B) 淡墨色彩消光，不反射映管（DARK TINT-ED, NON GLARE）为使电脑使用者在近距离，长时间观看荧光屏，它的荧光屏玻璃加有墨绿色素，以减轻眼睛疲劳现象，同时将光滑的玻璃荧光面腐蚀成锯齿状（或喷上砂质涂膜）以消除灯光反映在玻璃屏上。

(C) 点距（DOT PITCH）：一般来说，在文字处理上整个荧光幕可以显示 2,000 个字（行 $80 \times$ 列 25），在彩色显示器上，以最早期 IBM 的 CGA 来说明，它每个字占用的像素如图 2，为 $8 \times 8 = 64$ 个，其显示方式如图 1：

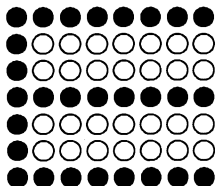


图 1 显示“E”的状况



每个图素中，有红，绿，蓝三个荧光点，变化素图中的三个荧光点亮度，即可得到无数种不同的色彩

图 2

因此如果每行要显示 80 个字，则荧光屏由左至右横向需有 $8 \times 80 = 640$ 个素图方可满足它的需求，同理每列有 25 个字，则由上到下纵向需有 $8 \times 25 = 200$ 个像素方可，这时素图间的距离（点距）为 0.43mm 或 0.39mm 即可如图 3。但是依实际经验得到， 8×8 个像素所构成的文字略嫌粗糙，因此在 IBM 的 EGA，就将它提高到 9×14 ，这时，横向要 $9 \times 80 = 720$ 个像

素，纵向需有 $14 \times 25 = 350$ 个像素以上，因之 $0.43\text{mm} / 0.39$ 就无法胜任而必需将点距缩小为 0.31mm，当然像素的大小也必需随之缩小了。

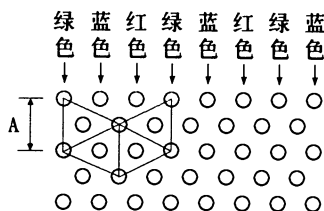


图 3

目前 38K 级显示器中 IBM 8514/A，VESA SVGA 已达到横向需要 1,024 个像素，纵向需要 768 个像素，于是，点距也必需缩小到 0.28mm，这也说明为何点距愈小，图像，文字愈清晰价格愈高的原因。目前，显示器业界由于成本的竞争剧烈，仍将 0.31mm 或 0.39mm 的显像管应用在 8514A/S VGA 上，并宣称它的分辨率（RESOLUTION）可达到 $1,024 \times 768$ ，其实严格的说来， $0.39\text{mm} / 0.31\text{mm}$ 显像管所能提供的横纵向像素（PIXEL），是无法达到如此高的数量，如何能分辨出 $1,024 \times 768$ 的解析度呢？充其量也只能说“能解读”（READABLE） $1,024 \times 768$ 这个 MODE 而已。

(D) 荧光材料：显示器的显示特性，是利用显像管里的电子枪（ELECTRON GUN）所发出的电子束（BEAM）打击在荧光屏上的荧光粉材料发光产生的。由于目前水平频率 31-38kHz 8514A/S VGA 所使用的频率范围其垂直频率为 50-70Hz（非隔行扫描）及 $1,024 \times 768$ 的 86.96Hz（隔行扫描），换句话说，它每秒钟所需变换的画面为 50-70 帧，（在 $1,024 \times 768$ 需变换到约 87 帧“半画面”），因此，在如此快的速度下，后一帧画面扫描完成时，前一帧画面必需很快消失，否则会造成后一帧画面遗留有前一帧画面有痕迹，即所谓文字或图片上的“拖尾”现象，因此荧光的选择，必需使用“余辉”时间短的材料，排列上这种材料属中短余辉的一级，故称中短余辉（M-S PERSISENCE）。

另在 35.5kHz 扫描频率的 $1,024 \times 768$ MODE 系使用隔行扫描的方式处理，也就是说它使用 1, 3, 5, 7 行……的奇数扫描的“半帧”画面与 2, 4, 6, 8 行……的偶数扫描“半帧”重叠在一起成为完整的一帧画面，（与目前彩电方式相同），其优点是要达到 $1,024 \times 768$ 的高分辨率使用的频率仅 35.52kHz 即可。成本低，设计简易，然而由于显像管属于中短余辉的特性，当后一帧“半画面”叠上前一帧“半画面”时，前一帧“半画面”扫

描已逐渐消失, 出现画面轻微的“闪烁 (FKICKER)”现象, 这就是为何显示器制造商一直努力使非隔行扫描 (NON-INTERACED) 做为显示器基本扫描方式的原因, 目前 1024×768 要达到垂直频率 60Hz (每秒变换 60 帧画面) 的水平频率为 48kHz。而 70Hz 时, 为 57KHz。

(E) 透光率: 一般约 57%, 由于荧光屏上加有墨绿色素, 使透光率降低, 加上不反射荧光屏表面处理, 使荧光屏的使用者视觉上改善很多, 唯一缺点是与会反射的营光幕 (GLARE TURE) 比较上有一些对比度 (CONTRAST) 稍差的缺点。

总体来说, 由于日本制造的显像管开发最早, 使用最多, 绝大部分的缺点都已经过实用改善过来, 因此目前以日制显像管品质最稳定, 各种特性最佳, 可靠性最好, 当然价格也比较贵。

(2) 同步信号: 一个电子画面的形成是利用电子由左到右的横向扫描, 将横向的, 荧光点一点一点地连接起来形成一条线如图 4, 再配合由上到下的纵向扫描, 将一线一线编排成为一个面。控制这些点连接与线编排之中的荧光点的亮或灭, 就可产生文字如图 5, 然而每一条扫描线的“起始点”, 要很规律地受到控制, 才能产生正确的显示画面, 否则将如图 6 一样, 产生歪扭的显示文字或图象, 因此需要同步信号 (SYNCHRONOUS SIGNAL) 来指挥。

(A) 波形: TTL (数字式), 取其准确, 稳定的脉冲性以控制显示器扫描的起始点以产生稳定的画面, 正极性或负极性 TTL 信号都可以准确地控制显示器画面。

(B) 水平频率: 一般 8514 / A SVGA 显示器适用范围为 31~38kHz, 控制方法有三种:

(a) 阶段式: 适用接口卡范围

- IBM VGA : 31.5kHz 640 × 350 / 400 / 480
- VESA S.VGA : 35.16kHz 800 × 600, 56Hz
- IBM 8514 / A : 35.52kHz 1, 024 × 768 (I)

(b) 半阶段式: 适用接口卡范围 (31~35k 阶段式, 35~38 全阶段式亦称复合同步信号式 (multisync))

- IBM VGA : 31.50kHz 640 × 350 / 400 / 480
- VESA S.VGA : 35.16kHz 800 × 600, 56Hz
- VESA S.VGA : 37.88kHz 800 × 600, 60Hz
- VESA S.VGA : 37.8kHz 640 × 480, 72Hz
- IBM 8514 / A : 35.52kHz 1, 024 × 768,

86.96Hz 隔行扫描

(c) 全阶段式: 适用接口卡范围: 31~38kHz 所有接口卡。(即 multisync 方式)

(C) 垂直频率: 8514 / A S.VGA 显示器一般适合的垂直频率为 50~90Hz。

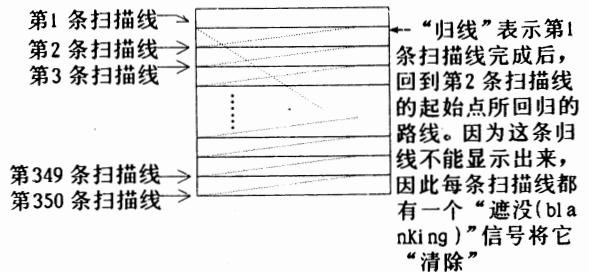
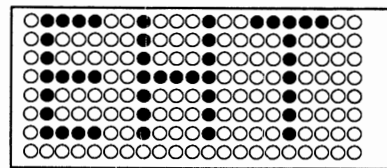
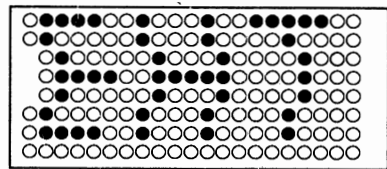


图 4 由扫描线构成画面的情形



受同步信号控制规律扫描起始点的画

图 5



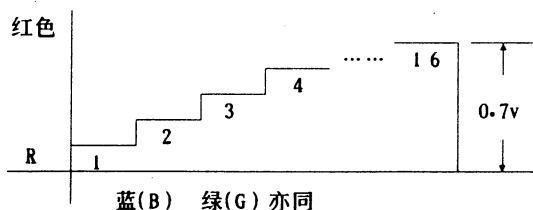
同步信号控制不良的扫描起始点不规律的画面

图 6

(3) 视频信号: 提供显示器用的视频电子信号, 电脑输出的信息, 全部包括在这个信号里面, 由显示器显示出来, 反之信息输入时也同。

(A) 波形: IBM VGA 的视频信号采用模拟式 (ANALOG), 有别于数字式。早期 IBM CGA 及 IBM VGA 使用数字式, 其优点是简单, 稳定, 不受干扰, 但缺点是彩色显像管只有红 (R), 绿 (G), 蓝 (B) 三只电子枪, 每只电子枪有一个数字波输入, 再怎样变化依排列组合, 画面仅能产生 $2^3 = 8$ 种颜色变化, IBM CGA 加入一个亮度变化, 也仅能产生 $2^4 = 16$ 种颜色, 相当单调, 因此在 IBM EGA 时, 利用网络将 R 分成 R, r, G 分成 G, g, B 分成 B, b, 也只能显示 $2^6 = 64$ 种颜色而已。因此在 IBM VGA 时将视频信号改为模拟式, 并将 R, G, B, 分割成 16 个位阶, 故能产生 $2^{16} = 4,096$ 种颜色, 但因 VGA 的 31~38kHz 范围的接口卡每个画面每次仅能输出 256 种颜色, 故显示器 4,096 种调色能力已绰绰有余。

(B) 波幅: 每个视频信号的波幅大小仅 0.7Vpp (0.7 伏特, 峰对峰), 在 0.7Vpp 里, 分割了 16 个位阶 (色层):



* 也就是说红色有深红、中红、浅红……等 16 种红色层, 绿、蓝亦同, 这样互相调配, 可调成 $2^{16}=4,096$ 色。

图 7

(4) 分辨率: 亦称解像度 (RESOLUTION), 即对一个画图的解析能力, 一般用横几个像素点 (PIXEL 纵几条线 (LINE) 来表示, 数目字愈大, 表示用更多的点及线来解析 (或分辨一帧画面, 当然就愈清晰了。目前 31~38kHz 接口卡皆包括以下这些模式 (MODE)。

(A) IBM VGA 有三个 MODE:

- 640×350: 为延续 IBM EGA 软件用
- 640×400 或 720×400: 文书, 文字处理用
- 640×480: 绘图功能用

(B) IBM 8514/A 增加一个 MODE:

- 1,024×768

(C) VESA S.VGA: 800×600 有二种规格:

- 800×600: 水平频率 35.16kHz 垂直 56Hz
- 800×600: 水平频率 37.88kHz 垂直 60Hz

(5) 显示颜色: 如 3- (B) 所述为 $2^{16}=4,096$ 种颜色

(6) 显像区域: 即荧光屏上能够显像的范围大小, 一般 IBM VGA 标准是横 240 毫米 (mm), 纵 180 毫米 (mm), 也就是 4:3 的横纵比例, 但一般设计是纵向 (高度) 可以自由调整, 横向 (宽度) 不能调整, 也有些显示器制造商亦将宽度做成可以调整的。同时 IBM VGA 的三个 MODE 里, 已将高度自动调整妥当, 变换这三个 MODE 高度是自动调整好的 (AUTO SIZING)。

(7) 兼容性: 表示显示器可以与电脑接口卡相通的能力, 一般市面上最普通的系统有 IBM VGA, IBM 8514/A, VESA VGA 等等。

(8) 输入电源: 表示显示器可使用的电源电压, 制造显示器时, 都会考虑到各国不同的电压系统, 如日

本 100V, 美国 110V, 欧洲 220V, 中国 220V, 澳洲 230V, 香港 200V……等等将其分割为二段, 即 110V 段与 220V 段, 110V 段显示器适用范围为 90~136V, 220V 段显示器适用范围为 180~267V, 有一个开关可以切换, 缺点是有时会因切错开关造成机器损坏现象, 因此高级的显示器就做成全域式电源, 即 90~267V 皆可适用。

(9) 功率消耗: 表示显示器消耗的电功率, 标示 85 瓦, 表示显示器最大消耗功率是 85 瓦, 显示器的画面范围在全白画面, (即全面点亮时) 功率消耗最大, 在这种情况下, 每 11.7 小时才消耗掉 1 度的电力, 因此它的耗电是很省的。

(10) 使用环境温度: 一般显示器是在室内工作, 温度都在 0~35℃ 之间, 也就是说在这样的温度下显示器的可靠性及平均故障间隔时间 (MTBF), 制造商是肯定可以满足自定的要求的, 因此制造商都定有保修期。

(11) 使用环境湿度: 在不结成水滴 (NON CONDESING) 的情况下, 在 10~90% 湿度下是可以安全使用的。

以上是一般显示器厂家规格标准, 读完它后可以让你判断这台显示器适不适合购用, 功能上是否可以满意, 再观察实物, 看看它的质量如何。

254

广告索引

- 1、广州白云山电源设备厂
- 2、广州易通计算机应用工程开发公司
- 3、广州市电子设备公司
- 4、赛宝星河
- 5、金泽科技实业有限公司
- 6、广州海谊电子仪器实业公司
- 7、华力科技开发公司
- 8、特强 (广州) 电子有限公司
- 9、华信电子系统公司
- 10、宏辉科技有限公司
- 11、南方计算机网络有限公司
- 12、广州袖珍计算机技术服务中心
- 13、宝迪电脑产品经营部
- 14、开拓电脑
- 15、香港现代电子出版社
- 16、高域电脑科技有限公司
- 17、广州军区后勤科技开发中心恒通综合经营部
- 18、华粤电脑工程公司
- 19、广东省计算机用户协会 AS/400 专业委员会
- 20、清华大学科学馆

283

条形统计的直接打印

甘肃省西峰市中国人民银行西峰支行 王有翦

【摘要】本文给出了一种把字符方式和图形结合起来直接打印条形统计图形的实现方法，并给出了完整的 C 语言实现程序。

一、问题的提出

在现在的 MIS 中，对统计图形的获得，一般都采用先在屏幕上显示，然后进行屏幕硬拷贝的方法得的，这种方法，存在着以下几个问题：

- 1、有关程序的编制困难，得到的程序可移植性差，不能适应各种类型的显示器的分辨率及色彩处理。
- 2、在分辨率低的显示器上得到的图形很粗糙。
- 3、不能充分发挥打印机的性能，得到的图形的尺寸有限。现在的 24 针打印机，每一行可以打印二千多个点，而 VGA 显示器的标准分辨率为 640×480 ，采用硬拷贝的方式，打印机的行宽利用了不到三分之一。即使在 1024×768 的方式下，也不过利用了二分之一不到，况且现在的许多系统软件都不支持这一方式。
- 4、得到的图形中的汉字只有一种字体，是十六点阵的显示字库的硬拷贝，很难看。

针对以上问题，本文给出了一种在 MIS 中直接打印统计图形的方法。

二、有关打印机知识

点阵打印机输出的图形，都是由许多离散的“点”组成，整幅的图形都是由若干的“行”组成，使用 24 针打印机时，采用不同的图形方式，一“行”图形由 24 个点“行”或八个“点”行组成，由于平常的打印机不能双向走纸，打印一般都要从上到下一行一行的进行。本文以 AR-3240 打印机为例，其打印针和送入打印机的数据的关系见参考文献[2]。

点阵打印机的图形输出，一般都要用到以下四组控制码：

- 1、图形方式控制码。用于通知打印机进入图形方式。
- 2、图形长度代码。通知打印机每一行打印多少个“点”。
- 3、图形数据代码。打印机输出的图形是由这些代码组成的。
- 4、行距代码。控制图形“行”间的距离。

通常，1 和 2 合在一起使用。有关进一步的知识，参见参考文献[1]。

三、原理、算法及数据结构

把一个条形图分解之后可以看到，条形图是由若干个单位长度的矩形加上单位长度的矩形的一部分组成。本文中的算法就是以此为基础的。把上述的单位长度的矩形再分解，可以看到它是由若干个以最大精度（图形能表示出的最小数值）为高度小矩形组成。这里的最大精度是指条形图所能表示出的最小的单位，如附图 1 所示。

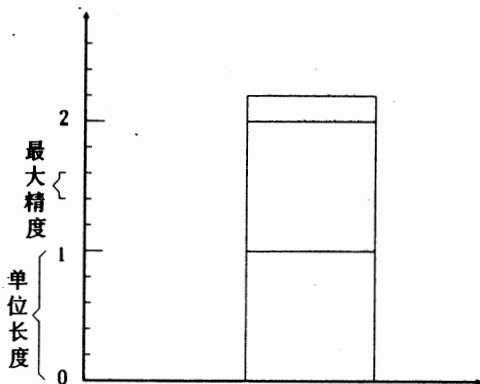


图 1

设要用条形图表示的数据为 $data[i]$, $0 < i < dn$, 其中 dn 为要处理的数据的项数。设 $maxd$ 为 $data[i]$ 中最大的一项，变量 k 为小于 $maxd$ 的 10 的整数幂中最大的一个，变量 k 的确定过程如下（用 C 语言表示）：

```
maxd = data[0];
for (i = 1; i < dn; i++)
    if (data[i] > maxd)
        maxd = data[i];
j = 0;
while (maxd > 10)
{
    maxd = maxd / 10; j = j + 1;
}
```

$k = \text{pow}(10, j)$
然后对 $data[i]$ 作如下处理：

```
for (i = 0; i < dn; i++)
    data[i] = data[i] / k;
```

这时 $data[i]$ 均不大于 10， $data[i]$ 的整数部分表示该

数据项形成的条形图含有几个单位长度的矩形。小数部分表示条形图顶端的那一部分，该部分的长度小于单位长度。

因为在不同的场合，对条形图的精度要求不同，本文及附程序中，都用变量 per 来控制，最大精度这 per 分之一单位长度。对 data[i] 作进一步处理，得到打印图形的有关数据，存放在结构指针变量 flag 中，flag 的定义如下：

```
struct f {int top, value;} * flag;
flag[i] 中的变量 top, value 由下列语句确定:
temp = (data[i] * per / 10 - floor (data[i] * per / 10));
if (! temp)
    flag[i].top = floor (data[i] * per / 10);
else
    flag[i].top = floor (data[i] * per / 10) + 1;
flag[i].value = floor (temp * per);
```

变量 top 存放条形图顶端的信息，value 存放条形图的高度信息。

数据处理完成后，逐“行”打印图形。先打印坐标的有关内容。再对 flag[i] 进行扫描，若满足条件就开始打印，把 value 的值赋为 100。

本文所附的例程中，per 的值为 5，一个单位长度由 5 个图形“行”组成，采用 24 针打印方式。有关的数字汉字采用字符方式打印，运行结果见附图 2。

四、进一步讨论

1. 和 DBF 文件的接口

这个问题在许多文献上已有论述，这里不再赘述。有兴趣者与作者联系。

2. 灰度精度和色彩

利用本文的思路，对于变量 C3 赋以不同的值可以得到不同的灰度效果，可以打樱蓼簇式、栈式条形图。若对数据处理更详细一些，可以将图示的精度达到一个“点”行，达到或超过屏幕硬拷贝得到的图形精度。

若需进一步改进程序，可以将彩色控制命令加入程序中，就可打樱蓼彩色条形图来。

本文是在 LC0530 微机上用 TURBO C2.0 和 AR-3240 打印机上实现的，例程及打印结果附后。

```
# include <stdio.h>
# include <math.h>
# include <string.h>
# include <stdlib.h>
void xyprint(int x,int y,char *buff,int count);
float data[12]={20,12.45,67,7.5,56,67,30,95,98,34,72,10,48};
char CR[2]={13,10};/*回车,换行*/
main()
{
```

```
char nu[11][4]={ " 0"," 1"," 2"," 3"," 4","
5"," 6"," 7"," 8"," 9"," 10"};
char tabname[20]="全年收入情况图";
char dw[11]="单位:20万元";
typedef char cc3[3];
char comm[5]={27,'*',33,0,0};/*图型方式命令*/
cc3 comm3={28,'3',48}; /*行距命令*/
cc3 comm4={27,'u','1'}; /*汉字字型控制命令*/
cc3 c3={0,0,0};
cc3 grp,pr={254,254,254},dot={128,0,1};
```

```
int i,j,k,dn,s,p,per;
char BLANK[4]=" ";
float maxd,temp;
struct f { int top,value; } * flag;
int dwkd=48,jgkd=24;
dn=12;per=5;
flag=( struct f *)malloc(dn*sizeof(struct f));
maxd=data[0];
for(i=1;i<dn;i++)
    if(data[i]>maxd)
        maxd=data[i]; j=0;
while(maxd>10)
{
    maxd=maxd/10; j=j+1;
}
k=pow(10,j);
for(i=0;i<dn;i++)
{ /*处理数据*/
    data[i]=data[i]/k;
    temp=(data[i]*per/10-floor(data[i]*per/10));
    if(! temp)
        flag[i].top=floor(data[i]*per/10);
    else
        flag[i].top=floor(data[i]*per/10)+1;
    flag[i].value=floor(temp*per);
}
k=dn*dwkd+dn*jgkd+6; /*计算每行的点数*/
comm[3]=k-256*floor(k/256);
/*计算图型方式命令的参数*/
comm[4]=floor(k/256);
grp[0]=0; grp[1]=0; grp[2]=0;
```

```
fwrite(&comm3,sizeof(char),3,stdprn);
comm4[2]='4';
fwrite(&comm4,sizeof(char),3,stdprn);
xyprint(28,0,tabname,strlen(tabname));
comm4[2]='1';
fwrite(&comm4,sizeof(char),3,stdprn);
xyprint(1,3,dw,strlen(dw));
fwrite(&CR,sizeof(char),2,stdprn);
```

```
for(j=per;j>=1;j--)
```



```

{
    fwrite(& nu[j], sizeof(char), 4, stdprn);
    for(p=per; p>=1; p--)
    {
        if(p<per)
            fwrite(& BLANK, sizeof(char), 4, stdprn);
        fwrite(& comm, sizeof(char), 5, stdprn);
        fwrite(& pr, sizeof(char), 3, stdprn);
        fwrite(& pr, sizeof(char), 3, stdprn);
        if(! (p-per))
        {
            dot[0]=128; dot[2]=1;
            for(s=0; s<4; s++)
                fwrite(& dot, sizeof(char), 3, stdprn);
        }
        else
        {
            dot[0]=128; dot[2]=1;
            fwrite(& dot, sizeof(char), 3, stdprn);
            fwrite(& dot, sizeof(char), 3, stdprn);
            dot[0]=0; dot[2]=0;
            fwrite(& dot, sizeof(char), 3, stdprn);
            fwrite(& dot, sizeof(char), 3, stdprn);
        }
        for(i=0; i<dn; i++)
        {
            if(! (flag[i].top-100))
                for(s=0; s<3; s++)
                    c3[s]=254;
            else
            {
                if(! (flag[i].top-j))
                {
                    if(! flag[i].value)
                    {
                        for(s=0; s<3; s++)
                            c3[s]=254;
                        flag[i].top=100;
                    }
                    if(! (flag[i].value-p))
                    {
                        for(s=0; s<3; s++)
                            c3[s]=254;
                        flag[i].top=100;
                    }
                }
            }
            else
            {
                for(s=0; s<3; s++)
                    c3[s]=0;
            }
        }
        for(s=0; s<jgkd; s++)
            fwrite(& grp, sizeof(char), 3, stdprn);
        for(s=jgkd; s<jgkd+dwkd; s++)
            fwrite(& c3, sizeof(char), 3, stdprn);
    }
}

```

```

for(s=0; s<3; s++)
    c3[s]=0;
}
fwrite(& CR, sizeof(char), 2, stdprn);
}
}
grp[0]=192;
grp[1]=0;
grp[2]=0;
fwrite(& nu[0], sizeof(char), 4, stdprn);
fwrite(& comm, sizeof(char), 5, stdprn);
fwrite(& pr, sizeof(char), 3, stdprn);
fwrite(& pr, sizeof(char), 3, stdprn);
for(i=0; i<k-2; i++)
    fwrite(& grp, sizeof(char), 3, stdprn);
fwrite(& CR, sizeof(char), 2, stdprn);
}

/*在指定的位置打印字符*/
void xyprint(int x, int y, char *buff, int count)
{
    int i;
    if(y)
        for(i=0; i<y; i++)
            fwrite(& CR, sizeof(char), 2, stdprn);
    if(x)
        for(i=0; i<x; i++)
            fputc(' ', stdprn);
    fwrite(buff, sizeof(char), count, stdprn);
}

```

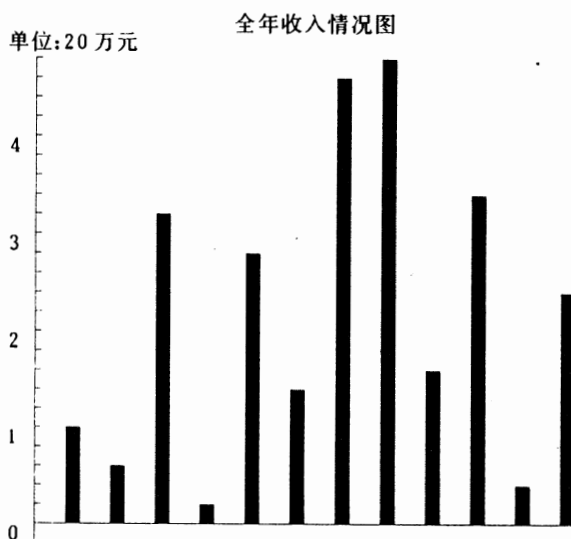


图 2

参考文献

- [1] 汉字系统和打印机参数 刘铁林等
《计算机世界》《明刊》 1990.12
- [2] 软件在修理打印机中的应用 田乃华
《微型计算机》 1992.3 (255)

四端网络幅频特性的自动测试系统

华南师范大学 李冰清

一、引言:

在模拟电路中, 电路的输出幅频曲线是电路的重要指标, 通常可以用信号发生器和电压表逐点描述, 也可以用扫频仪观察。随着计算机技术在各个领域的广泛应用, 测量仪器已经向着智能化的方向发展。通过接口电路把信号发生器、电压表和计算机连接起来, 由计算机控制信号发生器的输出信号, 并接收电压表的输出数据, 经过处理并打印出幅频曲线, 这样就能实现自动测试。本文介绍的是陷波器的幅频曲线的自动测试系统, 对于测试其他四端网络的幅频曲线的自动测试都可以参照这个方法实现。

二、自动测试系统的组成

本系统是由挂在总线上的控者(微机)、听者(接受信息的设备)、讲者(发出信息的设备)组成的, 如图1所示。下面分别简述各部份的特点和作用。

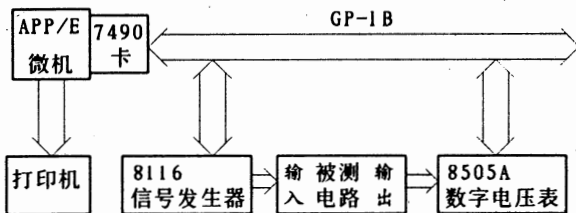


图1

1. 标准总线 GP-IB (General Purpose-Interface Bus)

标准总线采用国际电工委员会颁发的‘IEC-625-1号’和‘IEC-625-2号’标准。

GP-IB 标准总线由 24 条导线构成: 16 条信号线; 其中 8 条数据总线、3 条联络总线、5 条管理总线。另外 8 条是地线和屏蔽线。

8 条数据线传送 8 位数据。

5 条管理线传送以下五种信息:

① ATN (Attention) 线。ATN=1 时, 控者通过数据总线发出接口消息; ATN=0 时, 讲者通过数据总线发送仪器消息。

② IFC (Interface Clear) 线。当 IFC=1 时, 所有挂在总线上的仪器被清零复位。IFC=0 时, 清零取消。

③ REN (Remote Enable) 线。控者使挂在总线上的设备处于远地控制状态。

④ SRQ (Service Request) 线。由设备向控者发出请求服务消息。当 SRQ=1 时, 控者中断当前的工作, 响应服务请求。

⑤ EOI (End or Identify)。此线控者和讲者都可使用, 当 ATN=1 时, 控者发消息, 这时若 EOI=1 表示系统中控者正在进行点名识别; 当 ATN=0 时, 讲者发消息, 这时若 EOI=1 时, 表示讲者已发完数据, 给出结束信号。

3 条联络总线, 为保证系统能准确地传送数据, 采用三线挂钩技术。三线的功能分别是:

① DAV (Data Valid), 由控者或讲者向听者发送的数据有效信息, DAV=1 时, 数据有效 DAV=0 时, 数据无效。

② NRFD (NOT Ready For Data), 由听者发消息, 当 NRFD=1 时, 表示听者未准备好接收数据; 当 NRFD=0 时, 表示全部听者都准备好接收数据, 讲者或控者可发送消息。

③ NDAC (NOT Data Accepted), 由听者发出消息, 当 NDAC=1, 表示系统中至少还有一个听者未接收到数据, 源方应继续保持数据线上的数据有效。当 NDAC=0 时, 表示系统中全部听者都已接收完数据, 源方可以令数据无效。

2. 作为控者的计算机和与其适配的接口卡。

本系统采用 Apple-II 微机作为控者。进入 GP-IB 系统 Apple-II 用 7490-IEEE488 接口卡。IBM 微机可用 GPIB-PC 卡。本系统采用 7490 卡, 该卡插入第 3 扩展槽, 卡上的 25 针标准插座与 GP-IB 总线连接。

7490 卡提供了几种控制语句, 可被 BASIC 语言调用。下面介绍程序中使用的几个语句:

① 接口初始化语句: PR#3; 执行此语句后, 3#槽内的 7490 卡被选中。

② 消息发送语句:

PRINT"@地址码: '程控码 1、程控码 2.....'"

所表示的意义是: @表示使 ATN=1, REN=1 被控设备进入远程控制状态, 并准备好接收接口消息。紧接着的是受控设备的地址码由对应的 ASCII 码表示。单引号'.....'内表示的是仪器消息, 由仪器本身的功能决定, 其代码由仪器的生产厂家提供。以本程序中 80 行语句为例:

```
80 PRINT @ 0: ' W1M1FRQ1KHZ
FSOVAMP3.0VDO''
```

单引号中的内容即为仪器消息，这里是信号发生器 8116 的程控码，W1-正弦波，M1-常态，FRQ2KHz-频率为 2KHz，0FSOV-偏移量为 0V，AMP3.0V-输出幅度为 3.0V，D0-开输出。

③ 接口清除语句：

```
PRINT "%置 IFC=1
```

```
FOR A=0 TO 80: NEXT A 延迟约 100ms
```

```
PRINT "&"置 IFC=0
```

执行上述语句可产生一个宽度约为 100ms 的 IFC=1 的信号，使接口清零。许多 488 接口不需要这样宽的信号就可以清零，因此延时向视具体仪器而定。

④ 消息接收语句：

INPUT A\$；将由被寻址的讲者器件送出的数据读入 A\$ 中。

3. 带 GP-IB 接口的测试仪器

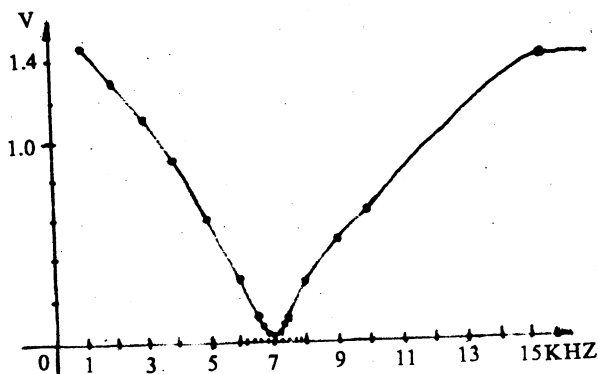
带 GP-IB 接口的仪器在仪器的后板上都有编程开关，该编程开关为 5 位，即可编五位二进制码。在五位地址码的前面还有两位为听、讲识别，01 为听 10 为讲。所以接口地址为听地址 01K5K4K3K2K1，讲地址 10K5K4K3K2K1。本系统信号发生器 HP8116 型为听者。其地址选 10000，听命令的 ASCII 码为 0；数字多用表为 FIUK8505 型其即是听者，也是讲者，本系统选其地址为 00001，听命令的 ASCII 码为！，讲命令的 ASCII 码为 A。

下面给出陷波器的幅频曲线的测试程序清单和测试结果。

```
LIST
10 DIM V(19)
20 FOR I=1 TO 19
30 PR# 3
40 PRINT "%
50 FOR J=0 TO 80: NEXT J
60 PRINT "&"
70 ON I GOTO 80,90,100,110,120,140,190,220,230,240,250,
260,270,290,300,310,320,330,340
80 PRINT "@0:WIM1FRQ1KHZOF50VAMP3.0VDO"
85 GOTO 430
90 PRINT "@0:WIM1FRQ2KHZOF50VAMP3.0VDO"
95 GOTO 430
100 PRINT "@0:WIM1FRQ3KHZOF50VAMP3.0VDO"
105 GOTO 430
110 PRINT "@0:WIM1FRQ4KHZOF50VAMP3.0VDO"
115 GOTO 430
120 PRINT "@0:WIM1FRQ5KHZOF50VAMP3.0VDO"
125 GOTO 430
140 PRINT "@0:WIM1FRQ6KHZOF50VAMP3.0VDO"
145 GOTO 430
190 PRINT "@0:WIM1FRQ6.5KHZOF50VAMP3.0VDO"
195 GOTO 430
220 PRINT "@0:WIM1FRQ6.8KHZOF50VAMP3.0VDO"
225 GOTO 430
230 PRINT "@0:WIM1FRQ6.9KHZOF50VAMP3.0VDO"
235 GOTO 430
```

```
240 PRINT "@0:WIM1FRQ7.0KHZOF50VAMP3.0VDO"
245 GOTO 430
250 PRINT "@0:WIM1FRQ7.1KHZOF50VAMP3.0VDO"
255 GOTO 430
260 PRINT "@0:WIM1FRQ7.2KHZOF50VAMP3.0VDO"
265 GOTO 430
270 PRINT "@0:WIM1FRQ7.3KHZOF50VAMP3.0VDO"
275 GOTO 430
290 PRINT "@0:WIM1FRQ7.5KHZOF50VAMP3.0VDO"
295 GOTO 430
300 PRINT "@0:WIM1FRQ8KHZOF50VAMP3.0VDO"
305 GOTO 430
310 PRINT "@0:WIM1FRQ9KHZOF50VAMP3.0VDO"
315 GOTO 430
320 PRINT "@0:WIM1FRQ10KHZOF50VAMP3.0VDO"
325 GOTO 430
330 PRINT "@0:WIM1FRQ15KHZOF50VAMP3.0VDO"
335 GOTO 430
340 PRINT "@0:WIM1FRQ20KHZOF50VAMP3.0VDO"
345 GOTO 430
430 PRINT "@!:VAR2S2F?"
440 PRINT "@A:"
450 PR# 0
458 PR# 1
460 INPUT V(I)
470 PRINT "V(I)=";V(I),
480 PR# 3
490 PRINT "@!: "
500 FOR K=0 TO 1000:NEXT K
510 NEXT I
520 PR# 0:IN# 0
525 STOP
530 HGR
550 HCOLOR=7
560 FOR M=1 TO 19
570 V(M)=INT(V(M)*50+0.5)
580 HPLOT 40+M*5,120-V(M)
590 NEXT M
591 IF N=1 GOTO 560
592 POKE 1913,1
593 PR# 1
594 PRINT CHR$(17)
595 PR# 0
596 N=1
600 GOTO 560
```

?+01.45576E+0	V(I)=.59592	V(I)=.03477	V(I)=.28238
V(I)=1.45576	V(I)=.29485	V(I)=.04355	V(I)=.52685
V(I)=1.3326	V(I)=.14635	V(I)=.07082	V(I)=.7569
V(I)=1.13838	V(I)=.0638	V(I)=.09856	V(I)=1.40192
V(I)=.88636	V(I)=.04312	V(I)=.15278	V(I)=1.40074



通用动态表格处理系统SJ-TAB

东南大学计算机系 沈 军

【摘要】SJ-TAB是一个新颖的表格处理系统。与现有的表格处理软件相比较，其最特出的一点就是动态性，即在数据填充阶段能根据实际数据的大小动态自动调整整个表格而无需修改原描述，从而提高了表格的适应能力，减少表格生成时间。因此，可以大幅度提高表格处理的效率，有较大的推广使用价值。本文介绍了SJ-TAB的设计思想、特点、系统组成以及各种功能。

一、前言

目前，为了减轻表格处理过程的复杂性，已研制出了许多表格处理工具，各有其特点。一般原理是，通过某种形式给出表格的描述，然后，内部将此描述自动转换成表格，或者是直接交互让用户通过键盘画出各种表格，在操作中提供一些自动检测和联想功能。无论是哪一种方法，一旦表格定型，在数据填充阶段就再也不能修改表格。除非重新修改描述，重新生成，重新填充数据。这样在数据量少，报表不复杂的情况下尚可。但在数据量大，尤其是报表复杂时对用户来说是难以忍受的。一方面表格生成周期拉长，另一方面增加了用户的负担。针对这一弊端，笔者提出了一种新的设计思想，即在第一部分生成的表格只是一个初表，在数据填入时可根据实际数据的宽高度自动地调整初表，直至数据填充完毕整个表才真正定型。可见，这是一个动态过程。整个过程中，不再需要返回去重新修改描述，重新生成，重新填充数据。从而，加快了表格生成周期，大幅度提高了表处理的效率，同时亦增加了表格的适应性。

二、SJ-TAB的特点

1、动态性

本系统是将整个表存放在缓冲区中，对表格的任意一条竖线可以左右移动，任意一条横线可以上下移动。同时，在数据填充时，根据实际数据的宽、高动态调整栏格的大小，随栏格的调整，整个表格亦自动作相应的调整。

2、描述简单

本系统定义的描述极其简单，只给出表格的形，而无需定义各种具体的量（如栏格宽度、高度等）。因为整个表是动态的，量的确定一直要到最后随实际数据而定。在描述中给出的量都是虚量。如不给出量，则系统取默认虚量。

描述分为2种，一种是树型表（规则表），见表1—表4，另一种是卡片表（非规则表），如卡片、简历表、登记表等。见表5。

表1:

描述为: (a, a, a)

←——
表头
←——

表2:

描述为: (a, b(b, b), b, b(b, b(b, b)), b)

	a ₁				b ₁			
						c ₁		
①	②	③	④	⑤	⑥	⑦	⑧	

←——
表头
←——

表3:

描述为: - (a, a (a, a (a, a)), a)

↑
表头
↑

表4:

描述为: (a, a (a, a), a (a, a (a, a))) /
—(a, a(a, a), a, a(a, a)a, a))

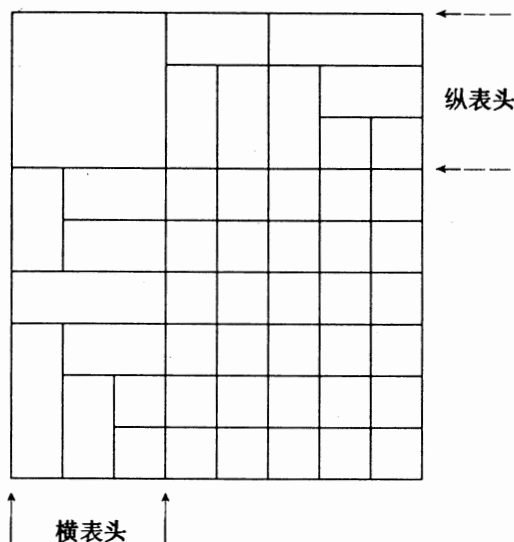


表 5:

描述为: #/姓名/10/年龄/4/性别/4/[8/
 #/籍贯//&/#/住址/14/电话号码//]/
 #/[/'个人简历'/
 #6/&/[/20/[/'#6/]/]/20/]/

姓名		年龄		性别		
籍贯						
住址		电话号码				
	个人简历					

注: 以上描述中都没有给出表头栏格宽度定义, 系统取默认虚值, 亦可给出予先的虚值, 如表 2 也可描述为: (a:3,b(b:4, b:2), b:5,b(b:4, b(b:10, b:20)), b:10)

其中的数字量对应于栏格①,②,③,④,⑤,⑥,⑦,⑧的预定虚宽, 而 a1, b1, c1 分别由②③,⑤⑥⑦,⑥⑦自动计算出。

3、快速原型

系统通过交互获取表格描述中, 快速自动生成表格框架原型, 若用户不满意, 可直接修改描述或直接进入编辑处理 (见第三、四部分), 增删、修改栏格形式, 直至满意。从而, 达到了快速原型化目的。

4、完全性

笔者在开发 MIS 过程中, 接触和分析了大量的表

格, 在此基础上提出了五种基本表原:

- a) 树型表 (见表 1, 表 2);
- b) 横向树型表 (见表 3);
- c) 纵横树型表 (见表 4);
- d) 卡片表 (见表 5);
- e) 万能表;

以及表头 (含表眉) 和表尾 (允许带有尾表) 编辑器, 对于 a), b), c) 可以一次成型。表头越复杂, 优越性就越显著; 效率越高。对于 d), 基本上一次成型, 亦可通过少量交互修改而成。e) 是一种交互稍多的全通用表。实际上是一个表格编辑器 (优于 CCED), 提供了对线的基本操作如添加一横 (纵) 线, 删除一横 (纵) 线, 移动一横 (纵) 线等。用它可以生成 a)~d), 但周期相对较长。因此, 一般能用 a)~d) 做的, 可直接用 a)~d)。提供 e) 只是为了使整个表格处理系统覆盖任意表格达到全通用, 实现 SJ-TAB 的完全性。

以上提出的只是表原操作, 任何一张大型复杂表格都可以拆成多个表原。因此, 对于大型复杂表一般要经过一次或多次自动拼接才能完成。

5、显示、打印任意长宽报表

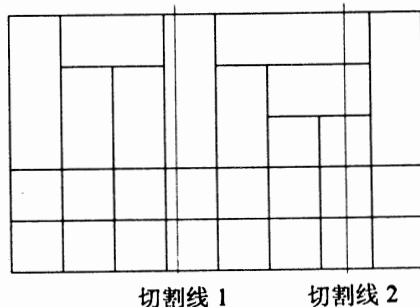
系统显示模块能显示任意长宽报表。上下、左右屏幅由用户设定, 系统提供上下、左右滚屏功能。

系统打印输出模块能处理任意长宽报表。其原理如下:

对于表 6, 其高度可由打印纸保证。纵向可任意长。而对于宽度, 一般打印机最大只有 132 字符宽, 对于大一些表显然不行。本系统采用切割方法, 可将宽表切割成几个窄表, 分多次输出, 最后人工拼贴一下即可。如此, 可处理任意宽报表。

表 6:

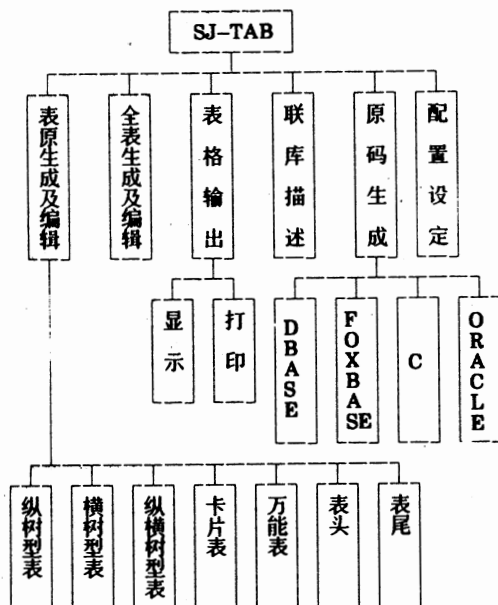
分三次输入



6、联库描述

表格可与数据库直接联接, 只要通过交互提供必要的描述即可。本系统与库的联接是在 SQL 级上。即由描述自动转换成 SQL 语言。因此, 可以联接多种数据库。另外, 对于规则表还能自动生成 dBASE、foxBASE+、C、oracle 源程序。

三、SJ-TAB 总体结构



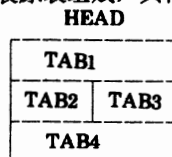
四、功能概述

1、表原生成

按照(二)中所给出的举例，交互输入描述串，系统自动生成各类表原表。

2、全表生成

由表原表名及有关算符按算符优先文法进行自动拼接，其中表头表尾拼接算符为“|”，上下拼接算符为“/”，左右拼接算符为“+”，而“()”可以改变优先关系。例如，[head|tab1/(tab2+tab3)/tab4|tail]表示一张大表由四个表原表组成，其构成图是



3、编辑 TALL

* 局部移线(上、下、左、右)

每次移动一个字符位置，移动时整个表不作相应调整，只是对指定线操作。

* 全局移线(上、下、左、右)

每次移动一个字符位置，移动时整个表自动作相应调整。

* 插入线(横线、纵线)

在用户指定位置插入横线或纵线

* 删除线(横线、纵线)

在用户指定位置删除某一横线或纵线

* 栏格编辑

由用户指定某一栏格后，采用多窗口技术，将该栏格内容送至编辑窗口进行编辑。在编辑时，允许用户新编辑内容的长、宽大于原来栏格，编辑退出返填时，系统将原栏格自动扩大，同时整个表亦自动作相应调整。

* 栏格复制

复制时允许源栏格和目标栏格大小不一致，系统自动作调整。

* 栏格传递

传递时允许源栏格和目标栏格大小不一致，系统自动作调整。

* 普通字符编辑功能

表头、表尾编辑器，除提供基本字符编辑功能外，还提供交互画表命令，是一个小的字表编辑器(因为有些表表尾可能含有尾表)

4、显示

每一栏格都可指定显示色(背景色、前景色)，有利于显示各种动态数据。

滚屏用的左右步长有：一个字符、一个特定步长、按栏格等。

上下步长有：一行、一屏。

5、打印

可指定各种字体

6、联库描述

交互指定某一栏格，然后给出联库描述，系统按此描述自动生成相应的 SQL 语句。

7、源码生成

对于规则表或卡片表(见表1表5)，可自动生成对应的源程序(dbase, foxbase, c, oracle)。

8、配置设定

可设定缓冲区(容纳整个表)的大小；各种颜色；步长；打印机型号等各种参数。

五、结束语

1、本系统的开发环境是：软件：C 语言，汇编语言。硬件：80286 及其兼容机。

2、本系统已用于银川地区电网自动调度系统工程和 MIS 系统的各种界面以及表格的生成，用户十分满意。

3、为了节省内存，本系统目前是在字符方式下开发的，因此对于照片一类的图形就不能填入表格，目前正在着手移植到图形方式下。

4、还需完善一些边界检测。

5、进一步开发提供多窗口、多表格操作，允许窗口之间表格的传递。从而各种表原可以在线操作。

6、进一步开发，将所有功能封装成库，提供一系列应用程序接口(API)，随用户任意调用。

高级 UNIX 连网技术讲座

广州昂立自动化工程公司 冯家宁

第二讲 TCP / IP (二)

2.5 传输控制协议

在上一节, 我们讨论了网际协议 Internet Protocol 如何提供一个虚拟的包交换网。连接到网际上的主机可以与任何连接到 Internet 的主机通信。本节集中讨论传输控制协议 Transmission Control Protocol (TCP) 这是一条转换协议, 它把面向低级数据报的 IP 服务 (其中数据报可能会丢失, 重复或乱序) 转换成一条全双工的、有可靠字符的数据流。

在最高一级, 应用程序常常要把大量数据从一台计算机发送到另一台。IP 的目的之一是提供强大的、易于使用的通信功能给程序员, 让他们写这些应用程序。一个方便的服务是可靠数据流服务。它使远程的应用程序收到的次序与发送时的数据流相同。数据流服务是容易理解的, 它能使许多应用程序有序地处理文件。

TCP 为运行在不同机器上的应用程序提供可靠数据流服务。它为应用程序提供无差错的、全双工通信通道。TCP 恢复丢失的包并保证数据的完整性。应用程序用 TCP 的 open 去连接远程应用程序, 收发数据, 当完成交换后, 就 close 连接。特别是一个应用程序可以作为一台服务器, 等待远程应用程序为它打开连接, 而另一应用程序作为客户, 它是通信的发起者。

TCP 把输入的数据流分解为小到可以放入 IP 数据报的段。IP 然后把各数据报携带到目的地的 TCP。为克服网际传输上的不可靠性, TCP 认可重发。每一节包含有序列号。一旦收到一段报文, 接收部份的 TCP 返回发送者一个认可, 表示段已收到。如果发送者在一段短时间内收不到认可, 它就认为数据已丢失并重发。

2.5.1 滑动窗口协议

TCP 属于一类叫滑动窗口协议的协议。窗口表示已被发送者发送的数据, 但接收者还未认可。当发送者收到认可, 窗口就前移, 表示有新的数据要传输。图 2-5a 表示一个滑动的窗口协议, 有 4 个包在发送者的窗口里。图 2-5b 表示发送者在收到对包 1 和包 2 的认可后的滑动窗口。

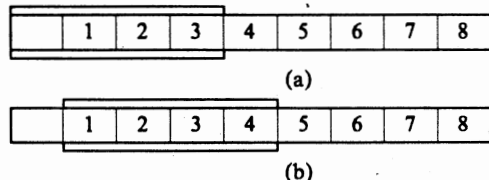


图 2-5 滑动窗口协议

滑动窗口协议的最简单形式叫“停-等协议”, 因为在某一时刻只能发送一个包。发送者发送一个单独的包, 然后停下来等待认可。“停-等协议”的缺点是它们不能完全利用网络的容量。例如, 多数网络可以在双向携带数据, 而“停-等协议”一个时刻只能在一个方向上发送数据。包首先传到目的地, 然后认可信息再回传。

为提高流量, 发送者增加窗口的大小, 使它能同时传送多个包。从概念上说, 每一个包都与其认可信息分开传送。“停-等协议”的流量就是一个来回时间一个包; 滑动窗口协议的流量是窗口的大小除以一次一个来回时间。

2.5.2 TCP 段格式

TCP 段格式由图 2-6 给出。象 IP 数据报, 段头有 20 字节固定的域, 跟随有一张或更多选择的表。4bit 的 offset 域指出段头在何处结束, 数据在何处开始。偏移量是以 4 字节为单位的。选择表放在段头的后部, 使段头长为 4 字节的乘积。段头并不包括一个长度域, 段的长度是由携带段的 IP 数据报的数据部份长决定的。其它各域在下面讨论。

1	9	17	25	32
源发端口				目标端口
序列号				
认可号				
偏移量	保留	控制	窗口	
核对和			应急指针	
TCP 选项				填补
...多达 65,515 字节的数据...				

图 2-6 TCP 段格式

2.5.3 端口

如应用程序要通信, 它们必须互相通名。即应用程序必须能指明给谁通信。从 TCP 的观点来看, TCP 软件必须能汇集到来的段。一个可能的做法是用两端点, 即目的地和源发地的 IP 地址。但 IP 地址是不够的, 因为我们想让多 TCP 连接到同一主机。TCP 用端口去解决寻址问题。一个 TCP 连接的每一个端有其 IP 地址和 16bit 的端口号标识。有一点要指出的是一个连接的两端必须唯一地址指定连接。

端口为服务命名。特别是客户在熟知的端口连接服务器。熟知端口是固定的, 极少改变, 确保它们作为程序的常量。进一步, 中央管理员为公认端口登记注册, 以致确保没有两项服务有相同的端口号。象邮件和远程注册等标准应用程序都使用熟知端口。

通常应用程序并不关心它用的端口号。特别地, 客户

打开连接后，只关心远端服务器的端口号。这样它就可以使用任何未使用的端口作为源发端口。但这并不意味着源发端口是不重要的。远程的 TCP 同时使用源发和目的端口号标识连接。的确，一个服务器可以用相同的本地端口支持无数个连接。因为连接将含有不同的 Internet 地址与端口号的组合。

2.5.4 序列号与段

应用程序把字节流交给 TCP，TCP 把它分解为小得可以放进 IP 数据项的段序列。为增加灵活性，TCP 把序列号与数据的每一个字节相联系而不是段本身。这样做的优点是有益于重发。如果有几个段丢失，TCP 可以把原来分开发送的几个段合并成一个段，减少重发次数。

TCP 使用 32 bit 序列号。序列号域在段头，给出段里第一个数据字节处在数据流的位置。当收到一个含有数据的段，接收部份的 TCP 返回一个认可。TCP 使用累积认可，它给出还未收到的字节最小值位于数据流的位置，而不只是说数据已收到。即认可域 (ACK) 是已到的数据的最高序列号。这样，如果一个段丢失，下一个段到达了目的地，返回的认可还是指向丢失的段，尽管收到了序列号较高的数据。

TCP 允许用选择的任何方式截断字节流，它也可以延迟发送数据以便把子序列的数据写入一个段。减少段数是重要的，因为它能降低处理数据的成本。同样，接收的 TCP 也可在把数据传给应用程序之前把数据缓冲，使数据在传给应用程序时能用一次操作有效地传送多段数据。

某些应用程序，特别是那些提供交互式的服务，一次只发送一个字符。如果 TCP 延迟发送交互数据，应用程序的响应将不可接受地变慢。TCP 为应用程序提供一种发送信号给 TCP 的办法来解决此问题。TCP 在收到信号后可以开放其缓冲区内的任何数据。信号机制是一个 push (推出) 选项，当应用程序写数据时用它。推出选项告诉 TCP 发送缓冲区里的所有数据。此外，发送的 TCP 在段头的 control 域设置 push 位，它强迫接收的 TCP 把任何缓冲区上的数据传给远程应用程序。由于推出选项并不提供指定记录边界的办法；它只是强迫把任何缓冲区上的数据送到远程应用程序。

2.5.5 建立连接

为支持通信的 client-server (客户-服务器) 模式，TCP 为应用程序提供两种打开连接的操作。服务器应用程序在想接受远程应用程序的连接时发出 passive opens 被动打开。客户通过发出指定远程服务器的 active opens 主动打开以连接服务器。TCP 用三趟握手建立连接 (图 2-7)。用段头控制域的 SYN 位建立初始序列号。握手的第一段在控制域设置 SYN 位。第一个 SYN 段并不认可任何数据。TCP 用控制域的 ACK 位表示认可域是有意义的。一旦连接建立 ACK 位是无用的。第二个段同时设置 SYN 位和 ACK 位；ACK 域承认收到第一个 SYN。第三个段承认收到第二个段。通常，客户打开与服务器的连接，服务器是先发出了被动打开的。但三趟握手的设计，

即使在两端同时发出主动打开的情况下功能也是正常的。

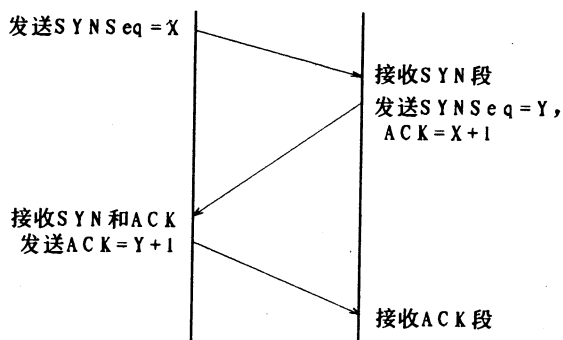


图 2-7 三趟 TCP 握手。

三趟握手是很重要的。首先，它保证了连接的两端预备好收发数据。其次，它建立了初始序列号。连接的每一端选择自己的初始序列号。序列号无需从 1 开始；实际上很少这样做。

2.5.6 结束连接

当两个应用程序已经完成交换数据时，它们 close 关闭连接。当一个应用程序告诉 TCP 设有更多的数据要发送时，发送端 TCP 设段头控制域的 FIN 位并告诉另一应用程序已无数据可发 (如：通过 UNIX 的 end-of-file 文件结束机制)。

TCP 连接是全双工的，它同时在双向携带数据。当发送端关闭一个连接时，它就禁止发送更多的数据。远端的 TCP 可以继续发送，但发送端必须继续接受数据，直到两端都发出连接关闭。只许单向流动的连接称为半打开的。

TCP 用 FIN 机制结束连接。但在某些情况下，一个连接不能关闭。如连接的一端主机出故障。主机重新启动后，所有有关 TCP 连接活动的状态信息都将丢失。如果收到一个前一次连接的段，接受主机将无法处理它。FIN 机制在此情形不适用，因为远程应用程序无法区别这是一个因线路故障而造成的连接中断还是远程应用程序已发出的关闭信号。TCP 用复位 (RST) 位让连接流产。只要收到一个与当前连接无关的段，TCP 返回一个设有 RST 位的段，当 TCP 收到此段，连接就终止。

2.5.7 最大段长选项

TCP 段只要能放进 IP 数据报，其长度是任意的。虽然 IP 可以携带多达 65515 字节的段，但许多计算机没有能处理这样长的段的资源。更重要的是，几乎还没有网络可以携带这样长的 IP 数据报。这样，IP 将分解大的段。表面上看，分解大的数据报是可行的。在实际上，它会引起大的问题。

一个例子揭示了网络接口设计的共同弱点。特别是某些接口不能处理一些帧之间间隔很小的“背靠背”帧的出现。即当第二个帧到来时，接口仍在处理第一个帧，第二

个帧因此而丢失。当 TCP 要发送一个大的段时, IP 可以把它分为较小的数据报。IP 发送快而连续的碎片, 导致网上背靠背帧的暴发。虽然 TCP 会最终重发未认可的段, 但事件序列不停地反复, 一个完整的段总到不了目的地。TCP 连接最终会超时。

TCP 用最大段长 (MSS) 选项以获得一个合适的段长。MSS 选项指出一个接收方愿意接受的最大段长。MSS 选项可以只当连接建立时用, 它只在 SYN 位设置的段里才是有效的。

2.5.8 流量控制

所有传输协议都要考虑的一个问题是流量的控制。端对端流量控制保证接收方有足够的缓冲区资源接收数据。如没有流量控制, 如发送方是超级计算机而接收方只是个人计算机, 发送方就很快填满接受方的数据接收区。

TCP 用流量控制窗口解决此问题。除了认可, TCP 段头包含有窗口广告域, 它指定接收方可以接受多少字节的数据。发送方 TCP 用接收方的广告窗口作为它的发送窗口。如接收方 TCP 数据缓冲区不够, 接收方 TCP 把广告窗口减少为零, 发送方就会停止发送。一旦接收方的应用程序把缓冲区的数据消耗掉, 接收方 TCP 就广告一个较大的窗口, 发送方又可开始发送数据。

2.5.9 核对和

TCP 段头有 16 bit 的核对和用于校验段头和数据。核对和是不一般的, 因为它包括 IP 报头的域。作为核对和计算的一部份, TCP 附加一个伪段头到它的段, 它含有 Internet 的源发和目的地址。伪段头的使用保证了段到达了正确的目的地。图 2-8 是 TCP 伪段头的格式。协议域是类型“TCP”, 长度域给出 TCP 段长。

1	9	17	25	32
源发 IP 地址				
目标 IP 地址				
零	协议	长度		

图 2-8 伪段头的格式

为计算校对和, 发送方 TCP 把伪段头附加到段, 用 0 补足段, 使它长为 8 字节的乘积并对全部结果计算校对和。如同其它 Internet 的校对和, TCP 用 16 bit 算法并取补的和的补。接收方 TCP 作与发送方相仿的计算, 从携带段的 IP 数据报里提取伪段头。

2.5.10 紧急指针

TCP 有一个应急机制, 它能使发送方应用程序告诉接收方应用程序接收紧急数据。发送方应用程序通知其 TCP 在数据流的哪一点上, 紧急数据结束。如果紧急指针标记了接收方应用程序没收到的数据, TCP 就告诉应用程序进入紧急模式, 应用程序处理紧急数据。TCP 用 16 bit 紧急指针通知接收方应用程序接收紧急数据。紧急域只在控制域的 URG 位设置后才有效。紧急域是序列号的偏移量, 紧急数据的实际位置由这两个域的相加来决定。

为了看看紧急功能的应用, 考虑一个在主机与终端之间携带交互信息的 TCP 连接。启动和停止输出的字符

(如: Control-S 和 Control-Q) 可看作是紧急的, 因为接收方要尽快处理它们。为提高效率, 紧急功能与推出机制一起用。

2.5.11 重发

TCP 最重要的思想之一就是接受方为已正确接收的段返回认可及对没得到认可的段进行重发。从概念上讲, 发送方在发送一个段时会启动一个计时器。一旦超时, TCP 就重发该段。一个有趣的问题是, 应取什么值作为发送者的重发超时计数 (RTO)。如果超时太短, 发送者就会重发只是延迟了而不是丢失的段。这就浪费了网络资源。如果超时太长, 发送者就会空等。较理想的重发超时值是与 RTT 的实际值接近。

为适应不同网络的不同延时, TCP 对当前往返时间 round trip time (RTT) 作动态估计。RTT 是发送一个段与接收一个认可之间所费的时间。由于 RTT 变化很大, TCP 取 RTT 的平均值, 平滑往返时间 smoothed round trip time (SRTT) 以减少异常长或短的 RTT 的影响。为跟踪长期的 RTT 的变化, SRTT 计算给较近的测量以较重的加权。这样, RTT 变化也引起 SRTT 的变化。以下是计算 SRTT 的算法

$$SRTT = (\alpha \times SRTT) + [(1-\alpha) \times RTT]$$
这里 α 是一个平滑因子, 它决定新度量的比重。当 $\alpha=1$ 时, 忽略新度量; 当 $\alpha=0$ 时, 新值取代旧值。典型的 α 值介于 0.88 与 0.9 之间。

SRTT 是对往返时间的估计。但不同段之间的 RTT, 由于排队和发送延迟是变化很大的。考虑到因延时造成的正常偏差, TCP 也通过期望值计算测量 RTT 的平均偏差。平均偏差 (MDEV) 是 RTT 和 SRTT 间的平均偏差。MDEV 提供标准偏差的近似值。如同计算 RTT, TCP 计算平滑值 (SMDEV) 如下:

$$SMDEV = (\alpha \times SMDEV) + [(1-\alpha) \times MDEV]$$

决定超时重发的最后值是

$$RTO = SRTT + 2 \times SMDEV$$

每当发送出一个段, 重发计时器设置一个 RTO 值。

当测量一个丢失或重发段的 RTT 时有一有趣的现象。发送方无法确定认可重发段还是第一次发送。如果不正确地假定是对第一次发送的认可, 测得的 RTT 值就会太大。如不正确地假定是对重发的认可, 发送方就低估了 RTT 值, 这会导致每段传送两次的浪费。早期的 TCP 存在这些问题; 新的 TCP 并不把重发段的 RTT 给 SRTT 作计算。

2.5.12 阻塞的避免与控制

TCP/IP 的设计目标之一是使它在广泛的网络种类上运行。的确, TCP 可以每秒上兆位地跨网运行, 也可在 1200 波特的低速网上运行。这种功能的结果是 TCP 必须动态地调整以适应变化的网络条件。在前一小节我们介绍了 TCP 调整以适应网络延时上的变化。在本节, 我们要介绍 TCP 如何调整以适应 Internet 路径上不同速度的变化。

只要底层允许,应用程序可用 TCP 取得超过每秒兆位的速度。TCP 通过产生和发送段以取得高速度。但如果 TCP 发送段太快,网关缓冲区一满,网络延时就增加了。最后网关变得超负荷以至无法理睬数据报,Internet 因而变得阻塞。

因为 TCP 是一个滑动窗口协议,TCP 可通过增加其滑动窗口的大小而增加流量,当网络阻塞时,发送方必须降低其传输率,这只要缩小窗口就能做到。

为做到控制阻塞,发送方 TCP 维护一个阻塞窗口。它不同于流量控制窗口,因为流量控制窗口是说明接收方能接受多少数据,而不说明底层网络的处理能力。阻塞窗口就指明底层可以携带多少数据。发送方 TCP 总是用两个窗口中较小的一个来决定发送的。

调整窗口以反映当前的网络条件是一项困难的工作。当应用程序打开一个 TCP 连接并开始发送数据时,其它共享网络的连接就会感到阻塞。当一个应用程序完成数据发送,网络的负荷就下降;其它应用程序应增大其窗口。

TCP 在它认为可增加带宽时慢慢地增加阻塞窗口而在它认为有阻塞时大幅度地压缩窗口。发送方增大窗口的速率是很慢的,这可以使网络常处于无阻塞状态。一旦检测到阻塞,TCP 就会立刻压缩其阻塞窗口。增大阻塞窗口的算法有两部份。当发送方从丢失一个段恢复过来时,慢起控制窗口的扩大。一旦发送方在网络路径上以正常的速度传送,控制开始转向避免阻塞。我们开始讨论阻塞的控制。

当发送方以网络路径容量或接近容量发送数据报时,突然增大窗口将会导致阻塞。但如果窗口大小一直不变,TCP 就不能在另一连接关闭时,充分利用网络的带宽。在计算避免阻塞的周期,无论发送方在什么时候收到新数据的认可,它都会增加阻塞窗口大小 $cwnd$ 的 $1/cwnd$ 。这一慢增量就使阻塞窗口在每一往返时间内增加一个段。这样,阻塞窗口的增加与时间成线性关系。

下一个问题是发送方如何检测阻塞。当一网关丢失一数据报时,它返回一个 ICMP 资源耗尽。甚至,该信号也会被丢失。为克服此问题,TCP 假定所有丢失的段都是因阻塞而丢失。当发送方的重发计时器一到时或它收到一个 ICMP 资源耗尽,TCP 就假定阻塞发生。

当发送方检测到阻塞,它就把阻塞窗口的大小减半并将其存于暂时变量 $ssthresh$ 中。然后它把阻塞窗口 $cwnd$ 设为 1 并开始慢起算法。在慢起过程中,无论它是否收到新数据的认可,发送方都把 $cwnd$ 加 1。慢起把窗口的大小按每一往返时间加倍。这样,算法的慢起部份按指数扩大窗口。一旦阻塞窗口超过 $ssthresh$,窗口扩大的算法回到阻塞避免上。

慢起对慢动窗口协议来说是共同的现象,那就是使用积累认可。只要有一段丢失,发送方就会重发并等待认可。当认可一到,它不仅认可了重发的段而且也认可了整个窗口的数据,开始一个新的数据窗口。如果发送方一次发送整个窗口,即使发送者只是重发一个丢失的段,网络

有可能阻塞并无法接收突发的数据报,这就增加了数据报被忽略的可能。慢起保证了任一次发送不多于两个段。

变量 $ssthresh$ 应予解释。当发送方重发一段时,它把 $ssthresh$ 设为阻塞窗口的一半。如果一网关时一数据报无反应,那一定是阻塞。在丢失段以前发送方 TCP 乐意以当前阻塞窗口的速率传输。当有新的连接开始传送时,网络阻塞很可能增加。这时,发送方会失去一半的带宽给新的连接。因而阻塞窗口也应该为原来的一半。慢起会很快把传输率恢复到段丢失前的一半。

慢起和避免阻塞在一个阻塞的网上相互协调。如果网上无阻塞,其性能会因为 TCP 假定所有丢失的段是阻塞所造成的而降低。这样,如果一个段在发送当中损失,TCP 会不必要地降低其速率。

2.6 用户数据报协议

并非所有应用程序都发现用面向可靠数据流的 TCP 服务是方便的。一些应用程序想用面向只了解记录边界的数据报通信。如分布式文件系统以文件为单位运行是很自然的。用户数据报协议 (UDP) 提供面向数据报的服务。UDP 提供无连续,不带可靠的传输服务,它用 IP 在机器之间携带报文。UDP 提供的比 IP 多些;它能在一台机器上区别多目的地并提供核对以保证数据的完整性。它不用认可信号也不对丢失的数据报重发。

与 TCP 不同,UDP 不是一个完整的传输协议。最重要的是它不提供阻塞和流量的控制。使用 UDP 的应用程序必须对此没有苛求。简单的说,它为应用程序提供了简单携带 IP 的最小功能。应用程序特别地用 UDP 作为更特殊协议的基石。在我们很快将要讨论的域名系统及 Sun 公司的 NFS 都是以 UDP 为基础的。

象 TCP,UDP 用 16bit 端口号标识数据报的目的地。UDP 端口号完全独立于 TCP 端口号,因为在 IP 数据报头里,TCP 段的类型域与 UDP 是不同的。图 2-9 示意 UDP 段头。长度域给出数据报报头和数据的总字节长。象 TCP,UDP 也附加一个伪段头给数据报作核对和计算。如图 2-9 所示,伪段头包含有 IP 的源发和目的地,IP 协议类型为 UDP,长为 UDP 数据报

1	9	17	25	32
源发端口		目标端口		
长度		核对和		

图 2-9 UDP 段头格式

2.7 域名系统

在前面各节里我们描述了 Internet 地址以及它如何用于 Internet 传输服务,为数据报选择路由。TCP 和 UDP 用伪段头的 Internet 地址计算核对和。在这一节里,我们介绍应用程序如何用域名系统去找与机器名相应的 Internet 地址。的确,几乎没人愿意直接使用 Internet 的地址,它们实在难记。

(作者地址: 广州东山金城宾馆曙前楼 511 室)

电话: 7754888-3511 邮编: 510080)

258

电子工业出版社广州科技公司

邮购书讯

MS-DOS 操作系统结构分析系列教材之一

• **BIO 结构分析教程** 郭嵩山编著

本书从操作系统原理和结构的角动, 采用以模块主程序为主线, 以数据结构为中心的系统软件分析方法, 彻底剖析了 MS-DOS (PC-DOS) 3.3 版操作系统的三大模块之一——基本输入输出模块。全书共分六章: IBMBIO 模块总体概述; IBMBIO 引导; 标准设备驱动程序; BIO 初始化实现原理; 系统初始化实现原理; 与系统配置文件的处理。

本书是由中山大学自然科学基金资助的有关操作系统研究科研课题成果之一。

本书力求从有利于教与学的角动对一般人感到难度很大的操作系统内部结构和实现原理通过深入浅出的系统论述, 让读者既能建立整体的概念, 又能逐步深入, 一层一层地剖析, 为了帮助读者能结合原理读懂程序清单, 对于程序清单的注释尽量详细, 力求深入到每一条指令。每章后均附有习题和思考题, 以帮助读者更好地理解 and 掌握。

本书可作为大专院校计算机有关专业的教材和教学参考书, 也是从事微型计算机系统和应用开发的工程技术人员常备的技术参考书。

全书 630 千字, 408 页, 16 开, 定价: 21.50 元
邮挂费 3.20 元 (电子工业出版社出版)

该系列教材之二三为:

COMMAND 结构分析教程; DOS 内核结构分析教程

均将在今年三、四季度出版发行

• **高频开关稳压电源** 张占松编著

本书系统介绍高频开关电源的工作原理, 电路构成、设计计算方法、实际使用的器件及调整方法。全书分 12 章, 从降压、升压变换器讲到它们的派生、组合的变换器; 从一般脉调制控制讲到准谐振、多谐振开关、零电流、零电压开关的控制; 从单端变换器讲到半桥、全桥变换器; 对开关器件 GTR、VMOSFET 等有专章分析、典型应用、调整方法有深入浅出的介绍。对兆赫级工作的谐振式零电压变换器, 多谐振变换器均有论述。书中有例题、习题。

全书 400 千字, 288 页, 16 开, 定价: 9.80 元,
邮挂费: 1.50 元 (广东科技出版社出版)

• **开关电源集成电路手册 (POWER INTEGRATED CIRCUIT DATA BOOK)**

该书系美国 POWER 公司有关开关电源 IC 芯片的英文选编本, 介绍的芯片有: PWR-SMP210、211、212、220、240、260 等, 以及以上芯片的应用资料, 包括: 利用开关 IC 的电路图, 印刷板图及有关高频变压器的资料。(注: 请读者注意, 该书为英文本)

全书 250 千字 202 页, 16 开, 定价: 18.00 元,
邮挂费: 2.70 元 (中国器材深圳公司资料)

• **MICROSOFT C6.0 大全**

徐为民 刘益敏等译

全书共分三篇: 分别为: “安装和使用专用开发系统, MICROSOFT C6.0 使用指南和 MICRORSOFT C6.0 高级编程技巧。本书是学习和掌握 MICROSOFT C6.0 的实用参考书, 适合广大从事计算机教学、研究和应用开发的科技人员使用。

全书 1100 千字, 723 页, 定价: 34.00 元, 邮挂费: 5.10 元 (中国科技大学出版社出版)

• **NOVELL 网络及其互联技术**

汤岳清编著 李智渊主审

目前, 国内计算机网络大都基于微机局域网 (PCLAN)。随着 NOVELL 网络的普及和应用的深入, 人们不再满足于单一的 NOVELL 网络在部门级充当文件和打印服务器的角色。特别是企业集团化使企业间通信要求大大增加, 这样, 把多个 NOVELL 网络和企业内的大型机、小型机或 UNIX 工作站以及与企业外的多种资源 (如 X.25) 连成一体, 达到共享资源和通信的目的。这是国内网络应用的现实课题和未来网络发展方向。

本书作者有承接多项网络工程的实践经验, 书中介绍了许多网络互联的技术问题。对 NOVELL 网络的工程开发具有重要的指导意义。

全书 250 千字, 16 开, 208 页, 定价: 13.00 元,
邮挂费 2.00 元 (电子工业出版社出版)

下期新书预告

- NOVELL 网络的安装、使用指南
- NOVELL 网络的故障维护技术
- 微计算机的下一场革命——多媒体技术

电子工业出版社广州科技公司邮购部地址:
广州市石牌华南师大北区一号 203

邮编: 510630 联系人: 王丽端

汇款请写明详细地址及邮编以便能准确及时的将您的邮购品寄出, 谢谢合作。

在真正的MS DOS5.0或DR DOS6.0下 运行王码5.0、金山5.0、2.13H等汉字系统

重庆市大足师范 宋运康

【摘要】文章分析了在MS DOS 5.0 DR和DOS 6.0运用流行汉字系统不能完全、充分发挥MS DOS 5.0和DR DOS 6.0性能的原因，给出了具体的实施方案，较好地解决了这一国内广大用户普遍关心的问题。

MS DOS 5.0和DR DOS 6.0都是目前在单用户个人计算机上运行的优秀的DOS操作系统。它们在继承以往的DOS操作系统的基础上，增加或增强了许多功能，特别是DR DOS 6.0，它使许多用户爱不释手。对于国内用户，大家最关心的问题显然是在新的操作系统下面能否使用流行的汉字系统。最近，已有不少刊物讨论在MS DOS 5.0下安装汉字系统的问题，但讨论的问题仅涉及与低版本的兼容性，只解决了在MS DOS 5.0下“可以运行”流行汉字系统的问题。但这种运行几乎是象征性的，因为它付出的代价似乎多于所得。对于在DR DOS 6.0下运行汉字系统的问题，几乎未见有讨论文章，这大概不是因为DR DOS 6.0比MS DOS 5.0推出较晚，在DR DOS 6.0下运行汉字系统，要付出的代价较MS DOS 5.0更多，几乎超出了用户可以承受的程度。

在MS DOS 5.0或DR DOS 6.0的众多优秀功能中，首推对于扩展内存的利用，它们把庞大的多个系统驻留文件移往640K常规内存之外，从而使用户较低版本DOS有更大的自由空间。但这个最主要的优点却与汉字系统的庞大字库相冲突，众所周之，容纳国标二级字库至少需要近240K空间。现在的情况是：当MS DOS 5.0或DR DOS 6.0只要有一个系统驻留文件进驻扩展内存，汉字库就只能占用常规内存，反过来，如果要使汉字库进入扩展内存，MS DOS 5.0或DR DOS 6.0的庞大系统文件就只能进驻常规内存。如是，MS DOS 5.0和DR DOS 6.0对于国内广大用户，似乎是爱之不能、弃之不忍的空中楼阁。

造成MS DOS 5.0或DR DOS 6.0的驻留文件与汉字库不能共处于扩展内存的矛盾，是因为两者都抢占扩展内存的下部（最靠近640K常规内存的部分）。上机检验可以证明这一点：一旦有一个MS DOS 5.0或DR DOS 6.0的驻留文件占用扩展内存，汉字库装载程序自动检测，认为无充足扩展内存，字库将被装入常规内存（如金山5.0）或留在硬盘（如王码5.0），若修改字库装载程序，强使字库进入原预定扩展内存位置，将明显感觉到系统程序被破

坏而死机。所谓“在真正的MS DOS 5.0或DR DOS 6.0下运行王码、金山、2.13H等汉字系统”是指：解决MS DOS 5.0或DR DOS 6.0的系统驻留文件与汉字库同时享用640K之外的扩展空间的矛盾，在和以往低版本DOS完全一样运行汉字系统的同时，完整、充分地发挥MS DOS 5.0或DR DOS 6.0的各种功能。解决的途径是要么修改MS DOS 5.0或DR DOS 6.0系统，要么修改汉字系统的字库装载和字模查找部分。显然修改后者的难度要低得多，且不会有任何后遗症。修改思路是：启动MS DOS 5.0或DR DOS 6.0的驻留文件进入扩展内存之后，强迫汉字库装入扩展内存中不与MS DOS 5.0或DR DOS 6.0的文件相重叠的位置，并修改汉字字模查找部分，使其能在新的位置中找到。确定新位置的起始处的最简方法是：先找到原字库装载起始位置，再在此基础上根据机器内存大小和需要向上方平移。笔者推荐把字库移到扩展内存的最上方，此时，可按如下关系计算：

平移距离 = 机器内存总容量 - 384K - 字库总容量 - 原字库安装位置起始地址。（这里的机器内存总容量按一般说法，如对于一般286机是1M，这里的384K是系统为显示缓冲区等用途保留）这里，需要简单介绍一下上述三个汉字系统是如何把字库装入扩展内存和如何在扩展内存查找所需汉字字模的。

2.13H的把扩展空间设置成虚拟盘，再把字库拷入，字库的使用同其它非虚拟盘上类似。它的缺点是需要虚拟盘管理程序。

王码5.0和金山5.0都是利用80286以上机的BIOS中断INT15的87、88号实现汉字库在扩展内存的存贮88号功能检测机器的扩展内存数量，入口参数是AH=87H，出口参数是AX=扩展内存数量（单位是K字节）。87号功能是在16M内存空间内（包含常规内存的640K）传递内存内容，调用前需要先设置一个称作全局描述符表（GDT）的参数块，该参数块由6个描述符组成，每个描述符为8个字节，结构如下：

00H	空描述字
08H	全局描述符表描述字
10H	源地址描述字
18H	目的地址描述字
20H	ROM BIOS 代码描述字
28H	ROM BIOS 栈描述字

与本文有密切关系的是第三、第四项描述符(即源地址描述符和目的地址描述符)的3-5字节(其他内容请参见有关资料),它们分别指定要移动的内存数据块的源地址和目的地址,要注意的是它们都是24位的线性地址(范围是000000H-FFFFFFH),不是通常的段:偏移量的形式,但仍是低位在前,高位在后。

再设置入口参数:

AH=87H

CX=传递数量(单位是字)

ES:SI=参数块首址

出口参数为:调用成功:ZF为1,AH为0;

调用失败:ZF为0,AH为错误码。

以下介绍用DEBUG对上述三个汉字系统的具体修改方法。并对有关片断作尽可能简略的注释。

(1)金山5.0:

debug chlib.com

-s100 51d0 cd 15 ; 查找int 15

1161:4A25

1161:4D5F

1161:4E16

-u4e16 4e1b

1161:4E14 B488 MOV AH,88

1161:4E16 CD15 INT 15

1161:4E18 3DEE00 CMP AX,00EE ; 检测是否至少有240K扩展内存

1161:4E1B 7202 JB 4E1F ; 若不足,跳转

-u49f4 4a25 ; 此段为从扩展内存取用所需字模

1161:49F4 81E27F7F AND DX,7F7F ; 屏蔽所取汉字内码的最高位

1161:49F8 81EA2121 SUB DX,2121 ; 计算该汉字的区位码(区号在DH,位号在DL)

1161:49FC 80FE09 CMP DH,09 ; 区号大于9?

1161:49FF 7203 JB 4A04 ; 否,跳转

1161:4A01 80EE06 SUB DH,06 ; 对于大于9的区号减6,跳过10-15空区

1161:4A04 8BC2 MOV AX,DX

1161:4A06 B05E MOV AL,5E

1161:4A08 F6E4 MUL AH ; 区号乘以94(5EH)

1161:4A0A 32F6 XOR DH,DH

1161:4A0C 03C2 ADD AX,DX ; 再加上位号;

1161:4A0E BA2000 MOV DX,0020

1161:4A11 F7E2 MUL DX ; 乘以32(20H),得到该汉字相对于字库头的位移量

1161:4A13 A31949 MOV [4919],AX ; 此位移量低16位送GDT源地址描述符相应位置

1161:4A16 80C210 ADD DL,10 ; 此位移量高8位加10H

1161:4A19 88161B49 MOV [491B],DL ; 此位移量高8位送GDT源地址描述符相应位置

1161:4A1D B91000 MOV CX,0010 ; 欲取16个字(32字节,一个汉字的字模)

1161:4A20 BE0749 MOV SI,4907 ; GDT首址送ES:SI

116 23 B487 MOV AH,87

1161:4A25 CD15 INT 15

-u4d3b 4d79 ; 此段为装入字库到扩展内存。

1161:4D3B 56 PUSH SI ; SI值为汉字库未装部分的容量,单位是节(16字节),第一次调用前为全字库容量3B7CH

1161:4D3C B90008 MOV CX,0800 ; 调用一次欲装入800H节

1161:4D3F 81FE0008 CMP SI,0800 ; 未装容量是否少于800H节?

1161:4D43 7302 JNB 4D47 ; 不少于,跳转

1161:4D45 8BCE MOV CX,SI ; 少于,则使欲装容量为剩余容量

1161:4D47 BAA851 MOV DX,51A8 ; 取装入缓冲区首址,(在常规内存)

1161:4D4A D1E1 SHL CX,1

1161:4D4C D1E1 SHL CX,1

1161:4D4E D1E1 SHL CX,1

1161:4D50 D1E1 SHL CX,1 ; 把该次欲装入容量换算成字节

1161:4D52 B43F MOV AH,3F

1161:4D54 CD21 INT 21 ; 读磁盘字库文件

1161:4D56 BE0749 MOV SI,4907 ; 取GDT首址

1161:4D59 8BC8 MOV CX,AX ; 传送字节数送CX

1161:4D5B D1E9 SHR CX,1 ; 换算成字数

1161:4D5D B487 MOV AH,87

1161:4D5F CD15 INT 15 ; 传送到扩展内存

1161:4D61 8106214908 ADD WORD PTR [4921],8000 ; 修改传送目的地址低16位

1161:4D67 8016234900 ADC BYTE PTR[4923],00 ; 修改目的地址高8位

1161:4D6C 5E POP SI

1161:4D6D 81EE0008 SUB SI,0800 ; 修改字库未装入量

1161:4D71 7204 JB 4D77 ; 若小于0,跳转4D77

1161:4D73 7402 JZ 4D77 ; 若等于0,跳转4D77

1161:4D75 EBC4 JMP 4D3B ; 若大于0,跳转4D3B,继续装

1161:4D77 B43E MOV AH,3E

1161:4D79 CD21 INT 21 ; 关闭磁盘字库文件

当跟踪到116c:4d5d处,查看原程序字库将装入扩展内存的起始位置:

-d4907 492f

1161:4900 00-00 00 00 00 00 00 00

1161:4910 00 00 00 00 00 00 00-80 A8 51 00 93 00 00 00

1161:4920 80 00 00 10 93 00 00-00 00 00 00 00 00 00

知原起始位置是100000H。此时可根据前述公式算得字库在扩展内存的新地址较旧地址的位移量,若机器内存为1M,则:

位移量=1M+384K-3b7c0H-100000H=024840H

现在可作如下修改:

-A51d1 51e0 ; 下列语句附加到chlib.com尾部,为修改GDT目的描述符中的数据传送目的地址初值,它将在程序驻留时和原程序的尾部一起被截掉,不致占用内存。

117D:51D1 PUSH BX

117D:51D2 MOV BX,4840

117D:51D5 ADD [4921],BX

117D:51D9 MOV BL,02

117D:51DB ADC [4923],BL

117D:51DF POP BX

117D:51E0 RET

-A2f30 2f42 ; 下列子程序为修改GDT源描述符中的数据传送源地址,将被多次调用,不能附加到原程序尾,笔者把它放入原程序中绝对安全的数据区,无副作用。

117D:2F30 PUSH BX

117D:2F31 MOV BX,4840

```

117D:2F34 ADD [4919],BX
117D:2F38 MOV BL,02
117D:2F3A ADC [491B],BL
117D:2F3E MOV SI,4907
117D:2F41 POP BX
117D:2F42 RET
-A100
117d:100 jmp 51d1
117d:103 jmp 4c30

```

最后用 R 命令修改 cx 中的文件长度为 50E1,用 W 命令存盘。

(2) WM5.0:

```
debug w5sk.wm5
```

```
-a100 15fe cd 15
```

```
1161:0E91
```

```
1161:112F
```

```
1161:1173
```

```
1161:140A
```

```
-u0c8f0c96 ; 此段检测扩展内存容量
```

```
1161:0E8F B488 MOV AH,88
```

```
1161:0E91 CD15 INT 15
```

```
1161:0E93 3D2C01 CMP AX,012C ; 判断是否至少有300K扩展内存
```

```
1161:0E96 722C JB 0EC4 ; 若不足,跳转.
```

```
-u1123 112f ; 此段装入字库, 每调用一次从常规内存中传送32K到扩展内存
```

```
1161:1123 E8F0FE CALL 1016 ; 涉及语句太多,不便注释
```

```
1161:1126 8D36E401 LEA SI,[01E4] ; 从ds:01e4单元取出GDT首址
```

```
1161:112A B487 MOV AH,87
```

```
1161:112C B90040 MOV CX,4000 ; 每次传送32K字节
```

```
1161:112F CD15 INT 15
```

```
-u1166 1173 ; 此段的作用仍为装入字库,这是WM5.0的特殊
```

```
1161:1166 E8ADFE CALL 1016
```

```
1161:1169 59 POP CX
```

```
1161:116A 41 INC CX
```

```
1161:116B D1E9 SHR CX,1
```

```
1161:116D 8D36E401 LEA SI,[01E4]
```

```
1161:1171 B487 MOV AH,87
```

```
1161:1173 CD15 INT 15
```

```
-u1400 140a ; 从扩展内存中取用所需要字模
```

```
1161:1400 07 POP ES
```

```
1161:1401 8D36E401 LEA SI,[01E4]
```

```
1161:1405 B487 MOV AH,87
```

```
1161:1407 B90002 MOV CX,0200 ; 取用512字节,而不是通常的32字节,是系统的又一特殊
```

```
1161:140A CD15 INT 15
```

首先修改 116c:0c96 处,以强迫字库装入新位置:

```
-a116c:0c96
```

```
116c:0c96 nop
```

```
116c:0c97 nop
```

再修改下列三处,以调用新增的子程序,使字库的装入、取用都到新的地址:

```
-a1126 1130
```

```
1161:1126 MOV CX,4000
```

```
1161:1129 CALL 1192
```

```
1161:112C NOP
```

```
1161:112D NOP
```

```
1161:112E NOP
```

```
1161:112F NOP
```

```
1161:1130 NOP
```

```
-a116d 1174
```

```
1161:116D CALL 1192
```

```
1161:1170 NOP
```

```
1161:1171 NOP
```

```
1161:1172 NOP
```

```
1161:1173 NOP
```

```
1161:1174 NOP
```

```
-a1401 140b
```

```
1161:1401 MOV CX,0200
```

```
1161:1404 CALL 11B3
```

```
1161:1407 NOP
```

```
1161:1408 NOP
```

```
1161:1409 NOP
```

```
1161:140A NOP
```

```
1161:140B NOP
```

下列子程序段使系统把字库装入新位置,由于此子程序将被连续几次调用(调用一次装入只能 32K),而原程序又自动修改每次调用时的目的地址(叠加 32K),所以下列子程序段在 int 15 之后再使目的地址还原,以抵消原程序中的叠加.(也可修改原程序目的地址叠加部分,但较繁).

再压缩位于 118a-11c8 处的一句较长的提示"Non library \$",为新增语句挤出安全空间: -e118a "no lib! \$" 然后增写如下程序段:

```
-a1192 11c8
```

```
1161:1192 LEA SI,[01E4]
```

```
1161:1196 MOV AX,4840
```

```
1161:1199 ES:
```

```
1161:119A ADD [SI+1A],AX ; 修改GDT目的地址低16位
```

```
1161:119D MOV AL,02
```

```
1161:119F ES:
```

```
1161:11A0 ADC [SI+1C],AL ; 修改GDT目的地址高8位
```

```
1161:11A3 MOV AH,87
```

```
1161:11A5 INT 15
```

```
1161:11A7 MOV AX,4840
```

```
1161:11AA SUB [SI+1A],AX ; 使GDT目的低16位还原
```

```
1161:11AD MOV AL,02
```

```
1161:11AF SUB [SI+1C],AL ; 使GDT目的高8位还原
```

```
1161:11B2 RET
```

下列子程序使系统在新的位置取得汉字点阵:

```
-a11b3
```

```
1161:11B3 LEA SI,[01E4]
```

```
1161:11B7 MOV AX,4840
```

```
1161:11BA ES:
```

```
1161:11BB ADD [SI+12],AX ; 修改GDT源地址低16位
```

```
1161:11BE MOV AL,02
```

```
1161:11C0 ES:
```

```
1161:11C1 ADC [SI+14],AL ; 修改GDT源地址高8位
```

```
1161:11C4 MOV AH,87
```

```
1161:11C6 INT 15
```

```
1161:11C8 RET
```

(3) 2.13 系统:

由于该系统是采用较落后的虚拟盘方式实现字库在扩展内存的存取,需要重新编写一段采用 INT15 的字库装入程序(不驻留),并修改或重新编制 INT7F 中断驻留程序(读显示字模中断).考虑到篇幅,不再给出具体程序.欢迎来信索取.

警惕CHKDSK报告的簇丢失及交叉错误

宁波东海微电子设计应用公司 徐永久

DOS 的 CHKDSK 命令功能是分析指定的或缺省驱动器上的目录、文件和文件分配表。并将校验结果生成一个盘状态报告。内容为：磁盘总空间、隐含文件所占空间、子目录所占空间、用户文件所占空间、盘上可利用空间、内存总容量、内存可用容量。

此命令能发现盘上丢失的分配单元(簇)，即根据目录表上文件位置对应的分配单元之外的分配单元。这些单元由不彻底的文件操作而产生，如在磁盘 I/O 操作时按了 Ctrl-Break 键。这些丢失的分配单元占据了有用的磁盘空间，可用“CHKDSK d: / F”命令发现丢失的簇。屏幕提示如下：

```
xxx lost clusters found in yyy chains
Convert lost chains to file (Y / N)?
```

提示用户在 YYY 链中找到 XXX 个丢失的簇，并询问是否把它们转换成文件。回答 Y 且使用了 F 参数后，将把丢失的簇对应数据恢复到形如 FILEnnnn.CHK 的文件中。

许多 DOS 用户在操作过程中，不太重视这个命令，很少应用它来查看或纠正自己计算机中的隐患。

本人在应用 Tango 3.16 画原理图时，发现昨晚用得好好的软件，今天早上一运行，机器就不停的鸣叫，键盘也被封锁，只好用复位解决。用 CHKDSK 一查，便发现 \TANGO 目录下有丢失的簇，这时不能冒然打一个 Y 了事。因为 CHKDSK 还报告了其他信息：

```
D:\TANGO\SCHEDIT.EXE
Is cross linked on cluster xxx 文件在第 XXX 簇交叉
D:\TANGO\SCHPLOT.EXE
Allocation error,size adjusted 文件分配有错，长度已调整好
D:\TANGO\XYJ001.S01
First cluster number is invalid,
entry truncated 文件的第一个簇号无效，已截断
```

一个比较安全的办法是不纠正错误，先把原理图文件(S01)和其他数据文件作备份，然后删除该目录下的所有文件，再从软盘上备份 Tango 3.16 系统，这样运行 SCHEDIT.EXE 时计算机就不会死机了。当然也可直接把软盘上备份的文件拷入该目录，但这样不能减少丢失的簇。一个很大的硬盘，用 CHKDSK 一检查免不了会发现这些错误，而且可能有几百个丢失的簇，如果用 F 参数，得到了几百个.CHK 文件，再通过用 TYPE 查看文件内容来恢复实在是得不偿失的，因此笔

者建议不如直接从软盘上拷贝。当然对于软盘没有备份的一些应用软件那就只好死马当活马医了。

笔者遇到的另一例是系统莫名其妙的不能启动。两个隐含系统文件 IO.SYS 及 MSDOS.SYS 都存在且首簇地址和文件长度也没有错，COMMAND.COM 文件也没有错误，用 FDISK 查分区表发现 PRI DOS 分区已经 Active，用 DM 也发现第一分区是可引导的，从软盘引导后，用 CHKDSK 便发现 IO.SYS 有交叉的簇，解决的办法是删除两个隐含系统文件，做一次 SYS 即可解决问题。还有一例不能启动的症状是把根目录下的 CONFIG.SYS 删除或改名后能启动，而恢复原来 CONFIG.SYS 文件的内容则死机。显然毛病在 CONFIG.SYS 文件里的设备驱动程序。排除某一设备驱动程序染病毒及长度不对的可能后，同样用 CHKDSK 发现该驱动程序有交叉的簇。

以上讨论了由于簇交叉引起应用软件及系统软件中的可执行文件不可执行的两个例子，当然对于软盘无备份的数据文件的簇交叉只好从.CHK 文件里来恢复。由于本人才疏学浅，不能对此命令有更深入的理解，希望本文能起到抛砖引玉的作用，得到广大同行及前辈们的指教。也算给同行及前辈们点一个题：

怎样防止簇的丢失及交叉而引起的数据丢失？

261

中华学习机维修一例

湖北省黄石教育学院微机室 王 曦

我室有一台 CEC-I 型中华学习机，开机后能进入西文 BASIC 状态，即出现西文 BASIC 状态下的提示符和光标。按空格键和字母键均正常，但按下回车键(Return 键)却显示字母 J，不能实现回车换行的动作；按下中文键却显示字母 M，不能进入中文 BASIC 状态，同时也不能进行自检。根据以上现象我分析极有可能是 KB3600 芯片出现了故障。因为从键盘键入的信息首先都要由 KB3600 对它们进行编码，显然该机直接收到从键盘键入的信息后，进行编码时出现了错误。更换一 KB3600 芯片后，再开机检测，故障现象消失。

262

DOS混装软故障初识

四川省涪陵地区卫生局 彭 禾

微机安装的 DOS 除版本高低, 还有众多提供该操作系统的公司之分, 诸如 MS DOS、(IBM) PC DOS、COMPAQ DOS、AST DOS 等。

DOS 版本高低为人们所熟知, 且版本不合每有明确的出错信息显示于屏幕, 一目了然。

在同版本号条件下, 各家公司提供的 DOS 主要命令及其格式基本一致。惟部份文件非原貌, 常见的是设备驱动程序易名改开关 (Switch), 例如: MS DOS 的 RAMDRIVE.SYS, 经协议单位 IBM、COMPAQ 等公司各自修改之后, 均另取名 VDISK.SYS。谨以 DOS V3.30 系列为例, 列其文件名及开关异同如下:

		Conventional DOS Memory	Extended Memory	Expanded Memory
MS DOS	RAMDRIVE	default	/ E	/ A
PC DOS	VDISK	default	/ E	-
COMPAQ	VDISK	/ B	/ E	/ X

(译名未统一, 扩充内存与扩展内存混淆, 暂用英文名)

此外, MS DOS V5.0 的磁盘高速缓存设备驱动程序—SMARTDRV.SYS, 在 COMPAQ DOS 中对应为 CACHE.EXE, 开关 (值) 也有改动。

在升级或换装其它版本 DOS 中, 不按常规操作, 原用版本的文件未删尽, 或未予覆盖, 每致高低版本、不同公司提供的 DOS 文件混装。此外, 当资料不全或难以掌握时, 用户往往另装习用的设备驱动程序, 也形成混装。

在某些情况下, DOS 混装无不良反应, 例如 COMPAQ DOS V 5.0CACHE.EXE 在 AST DOS V5.0 环境中运行正常。不过, 一旦在任何方面出现“不兼容”, 由于难以察觉, 尤其是文件名相同时, 每令人费解, 甚至错认为系统有误!

实践告示, 其对策有:

一、DOS 升级或换装必须按常规操作, 尽量准确使用安装程序。整套安装之前, 应删尽原用 DOS 文件, 杜绝混装。

二、升级或换装前应细读使用手册, 认识其异同, MS DOS 及 PC DOS 各种版本均有译本出售, 惟 COMPAQ DOS 需另阅随机手册—Reference Guide、User Program Reference 等, 方得要领。

实例一: AST PP3 / 33

故障现象

1、执行 FORMAT 命令失常, 或死锁, 或显示高容量软盘零道坏, 或 1.44MB 软盘仅得 720KB。(改用 PC TOOLS Initialize 功能均顺利格式化, 容量标准)

2、执行列目录命令显示内容不随所换软盘改变, 显示依旧。

原由探索

经销公司原装 COMPAQ DOS V3.31; 曾按常规升级为 COMPAQ DOS V5.0, 运行正常; 后得 AST DOS 5.0, 换装操作欠正规。

(两家公司提供的 DOS V5.0 文件混装, 部分命令不“兼容”)

处理方法

在排除软盘驱动器硬故障及计算机病毒感染之后, 删除 DOS 子目录内容, 用 AST DOS V5.0 (1.44MB×2) 按常规安装该系统。上述故障不复出现。

实例二: SJ 286 / 16

故障现象

凡 CONFIG.SYS 含建立虚拟盘命令 (下例), 启动微机即死锁。若删除该命令行, 启动及工作正常。

DEVICE=C: DOS\VDISK.SYS 384 512 / E

或

DEVICE=C: DOS\VDISK.SYS 128

原由探索

该机原用 PC DOS V3.30, 建立于 Extended Memory 384KB 虚拟盘工作良好; 新得 MS DOS V4.0 (复制格式), 遂用 PCTOOLS 删除硬盘 IBMBIO.COM 及 IBMDOS.COM, 从 A 驱启动, 以 A>SYS C: 传输两系统文件 IO.SYS 及 MSDOS.SYS, 并复制 COMMAND.COM 至 C 盘根目录, 再复制其余文件至原有的 DOS 子目录中, CONFIG.SYS 未触动 (含 DEVICE=C: DOS\VDISK.SYS 384 512 / E), 即后启动微机显示出错信息: Incorrect DOS version, 建立虚拟盘失败, 从外单位获得 COMPAQ DOS V4.0 VDISK.SYS, 覆盖原用的 PC DOS V3.30 VDISK.SYS, 启动微机即死锁, 又从另一台建有虚拟盘并在使用的微机上复制该文件, 再次装入, 仍死锁。

(MS DOS V4.0 不“兼容” COMPAQ DOS V4.0VDISK.SYS)

处理方法

从 A 驱启动, 列其 DOS 子目录查得 RAMDRIVE.SYS, 修改 CONFIG.SYS 中相应的命令行为: DEVICE=RAMDRIVE.SYS 384 / E, 再启动微机, 工作正常。

最后, 执行 C>DEL DOS, 删尽 DOS 子目录内容, 再将 MS DOSV4.0 全部文件复制到该子目录下。

(263)

一种查找LQ-1600K打印机故障的简捷方法

中国南方航空公司计算机中心 梁 军

对于富有经验的用户, LQ1600K 打印机设计了一个特殊的性能, 使之可以很容易的找寻出计算机与打印机之间的通讯故障原因。就是使打印机进入数据 DUMP 方式, 在此方式下, 传送到打印机的代码会被准确无误地打印出来。打印的格式如图 2 所示, 图中, 左边为计算机送至打印机的十六进制代码, 右边是这些代码相对应的字符。不可打印的代码, 如控制码等, 打印“.”为使打印机进入 DUMP 方式, 只要同时按下换页、换行按键, 再打开打印机电源开关。

为了便于查找故障, 我们可以用任何你熟悉的编程语言编制一段程序, 使计算机向打印机传送一系列有规律的字符到打印机打印在计算机输出之前, 先将打印机设置成 DUMP 方式。图 1 是用 DBASE 编制的程序, 它将使计算机传送出从 00-FF 十六进制代码至打印机。打印机输出如图 2 所示。从图 2 便可一目了然地看出计算机与打印机之间数据通讯的正确性。同时亦可看出打印机打印字符与代码对应的正确性。

当打印机数据电路有故障时, 例如, 由于某种原因

```
Data Dump Mode
01 02 03 04 05 06 07 08 09 0D 0A 0B 0C 0D 0E 0F .....
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F .....
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F !"# $ % & ' ( ) * + , - /
30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 0123456789:;<=>?
40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F @ABCDEFGHIJKLMNO
.....
```

图 2 在 DUMP 方式下, 打印机执行图一程序的结果

```
Data Dump Mode
01 02 03 04 05 06 07 08 09 0D 0A 0B 0C 0D 0E 0F .....
10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F .....
20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F ""$ % & ' ( ) * + , - /
30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D 3E 3F 0022446688::;<=>?
40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F @BDDFFHHJJLLNN
.....
```

图 3 在打印机接收 DATA0 恒为零时的字符输出

type ptest.txt

```
/!"# $ % & ' ( ) * + , - / 0123456789:;<=>?@ABCDEFGHIJKLMN O P Q R S T U V W X Y Z [\ ] _ ` abcdefghijklmnopqrstuvwxyz{ }- /
```

图 4 检查打印机输出字符正确性的文本文件

```
Data Dump Mode
0D 0A 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D ..!"# $ % & ' ( ) * + , -
2E 2F 30 31 32 33 34 35 36 37 38 39 3A 3B 3C 3D ./ 0123456789:;< =
3E 3F 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D >?@ABCDEFGHIJKLM
4E 4F 50 51 52 53 54 55 56 57 58 59 5A 5B 5C 5D NOPQRSTUVWXYZ[\ ]
5E 5F 60 61 62 63 64 65 66 67 68 69 6A 6B 6C 6D _ ` abcdefghijklm
6E 6F 70 71 72 73 74 75 76 77 78 79 7A 7B 7C 7D nopqrstuvwxyz{ }
7E 0D 0A
```

图 5 在 DUMP 方式下, 打印机执行 TYPE / PTEST.TXT 的结果

使打印机送至字符发生器的数据位 (即 DATA0) 恒为 0 时, 而主机送至打印机的代码是正确的。在 DUMP 方式下, 打印机输出如图 3 所示。

另外一种更为简单的方法是编辑一个内容如图 4 的文本文件。先把打印机设置成数据 DUMP 方式, 然后用 TYPE PTEST.TXT > PRN 命

令将该文本文件内容送至打印机。打印机将打印出图 5 所示的内容。用 ASCII 码表将图 4 和图 5 的内容进行对应比较, 即可找出错误所在。

附图:

```
x=0
set print on
do while x <= 255
  ?? chr(x)
  x=x+1
enddo
?''
?''
set print off
```

图 1 用 FOXBASE 编制的检查程序

如何快速更换

LQ-1600K 打印针

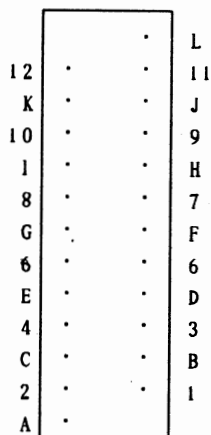
云南文山州人民银行科技科 王英伟

当发现 LQ-1600K 打印机出现漏针打印现象时,可用下述方法快速更换打印针,具体步骤如下:

1、用台钳夹住打印头外壳,用根木棒的一端顶住铜盖,一端用锤子敲击,抵出打印头(打印头外壳用沥青沾住),注意打印头快脱离时停止敲击,用手稍用力按出打印头。

2、用一字螺刀外撬三角钢套脱离打印头,取下铜盖,顺次将长针和短针取出,检查断针情况,配好完整的 24 颗针。

3、因 LQ-1600K 打印针分为两层排列,装针时较为困难,且不易判断长短针的位置,可采用如下方法装针:



出针排列图

长短针的位置如图,字母为短针,数字为长针,先将长针从 1 或 12 位置开始依次倒插入打印头,此时短针孔位已很明显,随意插入即可,完后再将倒插的长针轻轻拉出,这时长针孔位已很明显,随意插入即可。

265

CR3240 打印机

故障析修两例

西安武警技术学院光电系 杨轩举

例一、故障现象:色带时转时不转,打出的字迹时而清楚,时而模糊,甚至打印不出字迹。

故障分析:我们知道,打印机色带的转动是靠色带马达的转动带动色带架里的齿轮转动从而带动色带转动的。打印机其它部分工作正常而色带时转时不转,很可能是色带马达与逻辑主板的连接电缆 CN10C 有故障,拔下 CN10C,逐根用万表检查,发现电缆似断非断,接触不好。更换电缆后,故障排除,打印机工作正常。



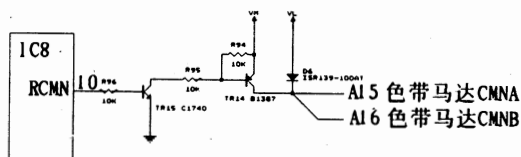
例二、打印机其它部分工作正常,但色带不转动,打不出清楚的字迹。

故障分析:

色带不转 (1) 可能是 CN10C 连接电缆断,拔下电缆,逐根检查,发现电缆完好疑点排除。(2) 色带马达损坏,用万用表检查马达完好,疑点排除。(3) 马达驱动电路故障。拆下打印机机械部分,检查逻辑板,发现色带电机驱动三极管 TR14B1387 损坏

故障处理:

TR14B1387 在市场上找不到此型号的管子,用 B1329 或 B1168 代换后,效果很好故障排除,打印机工作正常。



266

如何将四通电脑打字机上编辑 的文件在微机上使用

柳州市财政局 皮可慰

现在我国各机关、事业单位,使用“四通”系列中文打字机相当普遍,如果能将在四通打字机上编辑的文件在微机上也能使用,将是非常有益的。这是因为:微机上字处理软件丰富,能为编辑文件提供不少帮助;有时四通机安排不过来时,临时使用微机修改,也会感到不少方便之外。

四通机上的信息,可以用连接数据线的方式将数据传输到微机,但是往往两机相距过远,不少单位不容易办到。若能直接使用软盘在微机上处理是最为便利的了。

要达到这个目的,必须解决几个问题:

1、四通机使用 3.5 英寸软盘,微机也应该有 3.5 英寸软盘驱动器,好在目前不少高档微机上已配有 3.5 英寸软盘驱动器,为达到此目的提供了条件。

2、四通机软盘的格式化与微机的格式化有所不同。经过试验得知,大部分微机能接受从四通软盘拷贝过来的文件,而四通格式化的盘在微机上也能接受微机其他盘上拷贝过来的文件,但是必须非常慎重,否则会使得四通盘的信息丢失。

3、由于四通机中的中文内码与微机中的中文内码不同,必须进行转换才能在微机上使用,本人经过一段时间探索,用数据库命令编制了一个转换程序附后,供有兴趣者参考。

下面介绍该程序的使用方法:

1、准备工作:

(1) 在有 3.5 英寸软驱动器的微机的 C 盘内开辟 DBASE 子目录,并将 DBASE.EXE 和 DBASE.OVL 两个数据库程序拷贝入该子目录。

(2) 将本文后介绍的 STZH.PRg 文件输入该子目录。

(3) 在该子目录内建一个名为 ST.DBF 的数据库文件,只设一个段,段名定为 ST,属性是字符型,段长可定为 64(适应微机字处理的屏幕宽度,当然也可以更长一些),不需输入数据。

(4) 在 C 盘根目录下开辟 ST 子目录。

2、在四通机上对所须转换的文件作适当处理:

(1) 将文件的行长修改为 30 以下(如果前面所设的段长超过 64,则可以相应地超过 30)。

(2) 将自动换行符改为手动换行符,(因微机不识别自动换行符的内码)。

(3) 将自动换页符改为手动换页符,并且该手动换页符必须处于行首,因微机对自动或手动换页符均不识别,会将该换页符作为一个字符处理,如果换页符不处于行首,将会使行长超长,丢失信息。

(4) 将该不带后缀的文件,复制成一个带 TXT 后缀,并以英文为文件名的文件。

3、将带有经过修改并以 TXT 为后缀文件的四通盘,插入微机的 3.5 英寸软盘驱动器,进入 C:\ST 子目录,并拷贝该文件到该子目录,然后立即取出四通盘。注意只能对四通盘进行如此简单的操作,否则将可能破坏四通盘中的信息。

4、进入 C:\DBASE 子目录,并运行数据库。

在圆点状态下键:DO STZH

输入须转换的文件名(要带后缀 TXT)

运行结束,在 C:\ST 子目录下的该文件,即已成为微机可操作的文本文件了,可在多种字处理程序下修改、拷贝或打樱

5、如果希望将修改后的文件,再回到四通机上操作,将四通盘插入微机 3.5 英寸软盘驱动器,并进入该盘,用拷贝命令将 C:\ST 下的该文件,拷贝到四通盘上。注意拷贝到四通盘上的文件不应带后缀(因四通机上编辑的文本文件不带后缀),然后立即将四通盘取出,即拿到四通机上操作了。(注意在微机上,对四通盘只能进行如此简单的操作,否则可能破坏四通盘上的信息)

附:STZH.PRg 程序如后

```
clea
?
?
?
?
```

```

?
?
[ STZH.PRQ——四通转换]
[ ]
[ ]
[ 设计人:GXLZ-CZJ-PKW 1993.3.20]
?
[ ]
?[ 以下操作有误,可能破坏 A 盘信息!!!]
?
?[必须慎重将“A”盘四通文件复制到 c:\st\下,必须带 TXT 后缀 !!!]
?[ 原四通文件,不能带自动换行符,每行 30 字以下!!否则丢失信息]
?[如无,或无把握,键入“N”退出后,在 A:下用 DIR 检查.]
acce [ 确定进入键“Y”: ] to jr
if jr = "Y".or.jr = "y"
?
?[ 输入已复制到“C:\st\下”的四通文件名,带后缀 TXT.,回车后键 Y: ]
]
acce [ 记不清文件名,键 N 退出: ] to wjm
if wjm = "N".or.wjm = "n"
retu
endi
else
retu
endi
use st
zap
set cons off
appe from c:\st\&wjm sdf
go l
do while .t.
yz = trim(st)
xz = ""
yzcd = len(yz)
qs = 1
jz = subs(yz,qs,yzcd-qs+1)
do while .t.
jzcd = len(jz)
if len(jz) < 2
xz = xz+jz
exit
endi
jz1 = subs(jz,1,1)
jz2 = subs(jz,2,1)
if asc(jz1) < 160
xz = xz+jz1
jz = subs(jz,2,jzcd-1)
else
if asc(jz2) < 160
jz2 = chr(asc(jz2)+128)
endi
xz = xz+jz1+jz2
if jzcd-2 = 0
exit
endi
jz = subs(jz,3,jzcd-2)
endi
endd
repl st with xz
skip

```

```

if eof()
exit
endi
endd
set cons on
copy to c:\st\&wjm all sdf
use
retu

```

267

编辑先生:

您好!

我手头有个游戏名叫《勇者斗恶龙》是台湾精淘有限公司出品,容量约为 1M 左右,非常好玩。

便在进行战斗过程中,我找到了“风衣”、“回音之笛”、“银钥匙”、“洛特之剑”等法宝,但总是找不到最重要的“金钥匙”,致使游戏玩不下去。希望借贵刊一角,求得高手予以指点。(如需该软件,请来函索取。地址:浙江舟山市电力公司变电工区,邮编:316000)

致

礼

读者:傅剑

93.4.23

268

电脑编辑部全体工作人员:

您们好!

我从 1991 年开始订阅电脑杂志,本杂志是我的良师益友,对我在学习电脑知识方面帮助极大。现有一个问题想向您们请教一下:用什么方法可以得知每一个汉字在 16×16 点阵字库的 32 个字模,我参考其它杂志介绍公式“(区号)-1×94+位号”计算汉字在字库相应的地址位置查找,发现全不是该汉字的字模。我的电脑是 286 机,使用 CCDOS2.13H 及金山 5.0 两套中文系统,恳请您们在这方面详细介绍一下,是否可以用 BASIC 语言或其它语言编写一个程序:输入每一个汉字就能自动找出该汉字在字库的 32 个字模。

此致

敬礼

黄坚明

93.2.1

编者:请黄坚明先生留意本期宋运康先生的《在真正的 MS DOS5.0 或 DR DOS6.0 下运行王码 5.0、金山 5.0、2.13H 等汉字系统》一文。

269

控制程序执行次数的两种方法

江苏徐州中国矿业大学

周国均

程序设计人员时常面临着自己的程序被他人非法拷贝、使用的问题,除可采用设置口令或加解密技术外,还可通过控制程序执行次数的方法加以克服。笔者在平常编程实践中发现后者实现较容易,而且比较奏效。现将两种具体方法介绍如下:

一、日期控制法

在程序中首先调取系统日期(INT 21H 中 2a 号中断),判断其是否超过某一“极限日期”(例中为 1993 年 10 月 1 日),若则显示“软件失效”字样,而后进入死循环;否则进入程序主体部分,完成程序功能。

“软件失效”之对策有二:

1. 使用程序之前,先执行 DOS 的 DATE 命令设置系统日期,只要其值小于“极限日期”。

2. 打开源程序,重新设置“极限日期”然后再编译成可执行文件。

二、计数器控制法

将程序最多可执行的次数 COUNT 预先放入一数据文件(例子中为 COUNT.DAT),可以采用任何编辑软件进行编辑。为迷惑他人,扩展名可改为 EXE 或者 COM。

在程序中首先打开数据文件,读出 COUNT,判断其值是否小于 1,若则出现“软件失效”字样,进入死循环;否则继续执行程序主体部分。程序结束前,把 COUNT 减 1 后写入该数据文件中。这样每执行程序一次,数据文件中 COUNT 便减少 1,直至为零。

该方法在“软件失效”后的补救措施比较简单,只需打开数据文件重置 COUNT 即可。

笔者用 C 语言编写了实现上述两种方法的程序(上机调试均已通过),源程序如下:

1. 日期控制(datecon1.c)
#include "stdio.h"

```
#include "dos.h"
#define YEAR 1993
#define MONTH 10
#define DAY 1
union REGS in, out;
main ( )
{in.h.ah=0x2a;
 intdos(& in, & out); /*取系统日期*/
 if(out.x.cx>YEAR)
  illegal();
 else if (out.x.cx==YEAR)
  if (out.h.dh>MONTH)
   illegal();
  else if(out.h.dh==MONTH)
   if(out.h.dl>DAY)
    illegal ( );
 /*.....程序主体部分.....*/
 }
 illegal() /*提示软件失效,死循环! */
 {printf("THE software is out of date! \n");
  for(;;);
 }
```

```
2. 计数器控制(counterc.c)
#include "stdio.h"
int count;
main ( )
{FILE *in,*out;
 if ((in=fopen("count.dat","rb"))==NULL)
 {printf("Can not open file\n");
  exit(1)
 }
 fscanf(in,%d,& count);
 fclose(in); /*... 打开数据文件 ....*/
 if(count<=1)illegal();
 /*.....程序主体部分.....*/
 count=count-1;
 if((out=fopen("count.dat","wb"))==NULL)
 {printf("Can not open file\n");
  exit(1)
 }
 fprintf(out,"%d",count);
 fclose(out); /*关闭文件*/
 }
 illegal ()/*提示软件失效,死循环! */
 {printf("THE software is out of date! \n");
  for(;;);
 }
```

270

SOURCER4.56 安装号之拆解

华南理工大学 朱 彤

SOURCE4.56 是新一代的超级反汇编工具。它甚至能支持 80486 的保护模式;能生成 MASM5.0、MASM5.1、MASM6.0、TASM2.X、TASM3.0 以及 OPTASM 的汇编源程序,而且支持协处理器 (80X87) 指令。可以说 SOURCE4.56 是汇编级上程序剖析工作者的得力助手。

遗憾的是,当你兴冲冲地想试试自己的 SOURCE4.56 拷贝版时,它会让你输入 PRODUCT SERIAL NUMBER,因为 SOURCER 不是 SHAREWARE 的,其产品安装号写在商品盘上,是一串冗长的数字、字母及符号。

笔者用 DEBUG 对其进行了耐心的跟踪和详尽的分析,发现 PRODUCT SERIAL NUMBER 开头 8 位并非透明的。当你因无安装号而沮丧地退出时,你会发现在返回菜单的右上角有一些很令人兴奋字符,例如:S307620-XBIO。但不要高兴过早,因为这并不是完全的安装号,仅前 8 位是正确的,后 4 位怎么办?

经分析,发现其后 4 位虽然对用户来说是透明的,但也是采用某种算法,由前推后的。该算法如下:

```
MOV SI, 5A53 ;指向所输入的安装号
MOV AX, 0001
MOV DL, 07
MOV CL, 00
AND BYTE PTR[SI], DF ;屏蔽 BJT5 使大小写字母等效
S: MOV BL, [SI]
XOR BH, BH
PUSH AX
A: MOV AX, BX
SHL AX, CL
MOV BX, AX
POP AX
ADD AX, BX
TEST BYTE PTR[SI], 01
JZ B
SHL AX, 1
```

```
B: TEST BYTE PTR[SI], 02
JZ C
XOR AX, 55AA
C: INC CL
INC SI
DEC DL
CMP DL, 00
JNZ S ;对前 7 个 ASCII 码计算,结果存在 AX 中
INC SI
MOV BX, 5E08 ;内部密匙表
MOV CH, 04
MOV CL, 04
D: ROL AX, CL
MOV DX, AX
AND AX, 000F
XLAT ;变换
AND BYTE PTR[DI], DF
CMP AL, [SI] ;核对密码
* JNZ ERROR ;转出错处理
INC SI
MOV AX, DX
DEC CH
CMP CH, 00
JNZ D
```

仔细分析以上算法可推出全部安装号,但是其演算过程是相当繁琐的。在此笔者推荐一个简便的方法,以满足那些急切地想使用 SOURCER4.56 的用户。

用 PCTOOLS 中文件服务功能的“F”命令,搜寻十六进制数“75B2”,找到后,将它改为“74B2”,即把上述程序段中加“*”号的那一行,改为“JZ ERROR”。

这样,只要你简单地把 SOURCER4.56 已经给出的 8 个字符正确输入,再随意敲入 4 个字符,一切就全部完成。

不过,请注意:千万不要把后 4 个输正确了,否则,它会当作是你输错了安装号,而退出。所以,最好是随意地打入诸如:0000,FACE,LOVE,之类的字符。

好了,祝你成为 SOURCER4.56 的合法用户!

中华机固化DOS研究

重庆市白市驿 39940 部队 王建民

在中华学习机内固化 DOS, 是很多电脑爱好者非常感兴趣的事。虽然固化 DOS 可以采用扩展槽口外接 ROM 的方法实现, 但制作颇为不便, 而且会占据中华机仅有的一个扩展槽, 这对某些特殊用户来说是不能容忍的。如何固化好 DOS 而又不另外添置扩充硬件设备呢? 这就是本文同您研究的问题。

一、固化的 DOS 有哪些优点呢?

1、固化 DOS 后, 开机键入 DS, DOS 系统瞬间自动进入 RAM 内部自定位, 并设置好有关参数, 供你使用 DOS 命令, 减少了每次引导磁盘的磨损。

2、固化的 DOS, 不影响从其它软盘调入的 DOS, 即软盘引导 DOS 后, 只要不键入 DS 命令, 固化的 DOS 就不会侵入 RAM 内部。

3、固化 DOS 后开机即可方便地用 BASIC 编程, 引入 DOS 可在编程之前或之后, 避免了编程后未引导 DOS 或 DOS 遭破坏为保护 BASIC 程序而进行的繁琐手续。

4、固化 DOS 后, 可以腾出近 10.5K 的空间运行程序, 可以起到大卫 DOS 的功能而又不占用 16K 语言卡。在运行 BASIC 时可以设置 HIMEM: \$C000, 虽然运行程序时 DOS 遭到破坏, 但可用 DS 命令快速予以恢复且无任何不良影响, 从而间接地提供原来 DOS 占用的空间供你使用。如固化功能较强的其它 DOS 系统 (如大卫 DOS) 则性能更佳。

5、便于软件的解密。

二、中华机固化 DOS 的空间

要想固化 DOS, 必须寻找廉价的固化空间。CEC-I 是苹果机的增强型兼容机, 除了扩充部分 BASIC 解释命令, 增加了汉字等功能外, 还利用了体切换技术, 在辅存 ROM 的第二体内固化了一个 LOGO 语言子集。其中, 在 U7 内物理地址低 16K (\$9000-\$3FFF) 中固化的是这个 LOGO 子集的

部分内容。这 16K 空间来固化 10.5K 的 DOS3.3 是十分理想的。这是因为:

(一)、LOGO 语言在一般家庭中使用甚少, 这 16K 一直空间让其占据而又不使用, 实在是一种浪费。

(二)、DOS 固化中不仅要解决固化整体问题, 还要解决向量设置, DOS 移位命令的解释等工作。这些命令均设置在 U7 的高地址部分, 固化时仅需要改变一片 U7, 改动范围相对较小。

(三)、即使保留 LOGO 语言, 只需取下机的 U7 (中华机内 U7 都是装在集成块插座上的), 换上自己固化的 U7。一旦需要恢复原机功能, 只需将原来的 U7 装入即可。

(四)、固化 DOS 后, U7 内尚有 5.5K 空间未利用, 还可以略施小计, 固化自己感兴趣的程序。

三、怎样完成 DOS 的固化?

我们利用手中现有的与 CEC-I 配合的 EPROM 固化, 采取了一种为保守的手续, 按照修改 DOS—移置兼并—修改系统程序—存盘—固化的步骤, 很快便完成了固化工作。

(一)、修改 DOS;

1、用 DOS3.3 系统盘冷启动中华机。

2、修改 DOS3.3 中不尽人意的地方。

]CALL-151 ; 进入监控

* A964: FF ; 增加 BLOAD 的长度到 64K

* ADAC: 58 FC ; 使 CATALOG 先清屏后列目录

3、在 DOS3.3 文件缓冲区内设置需要的有关数据

9718-20	58	FC	EA	EA	EA	EA	A0
9720-00	B9	31	97	20	ED	FD	C8
9728-C0	46	D0	F5	60	A0	A0	A0
9730-A0	8D	8D	8D	8D	8D	8D	A0
9738-A0	A0	A0	A0	A0	A0	A0	A0
9740-A0	A0	A0	A0	C7	D5	A0	C8
9748-D5	CE	A0	C4	CF	D3	B3	AE

9750-B3	8D	8D	8D	8D	A0	A0	A0
9758-A0	A0	A0	A0	A0	A0	A0	A0
9760-A0	A0	A0	D6	C5	D2	D3	C9
9768-CF	CE	A0	B1	AE	B1	8D	8D
9770-8D	8D	8D	8D	8D	8D	8D	8D
9778-A0	A0	A0	A0	00	00	00	00
9780-A9	9E	85	39	EA	85	37	A9
9788-81	85	38	A9	BD	85	36	EA
9790-EA	A9	00	85	73	85	0F	A9
9798-96	85	74	85	70	EA	EA	A0
97A0-00	B9	B0	97	99	D0	03	C8
97A8-C0	30	D0	F5	EA	EA	EA	60
97B0-4C	BF	9D	4C	84	9D	4C	FD
97B8-AA	4C	B5	B7	AD	0F	9D	AC
97C0-0E	9D	60	AD	C2	AA	AC	CI
97C8-AA	60	4C	51	A8	EA	EA	4C
97D0-59	FA	BF	9D	38	4C	58	FF
97D8-4C	65	FF	4C	65	FF	65	FF

(二)、移置兼并:

1、先将 DOS 移到 RAM \$ 1000-\$ 8FFF 中,以便与新的 U7 物理地址 \$ 0000-\$ 7FFF 相对应。

* 1000<9600.BFFFF

2、因为固化 DOS 后,原系统固化的 BASIC 解释程序需要修改,我们可把它调入 RAM 相应的位置后再行修改。

* 6000<D000.FFFFF

3、在中华机内, U7 物理地址 \$ 4100-\$ 4FFF 映身内部 ROM 的内容,我们也需将它移入 RAM。

* C007: 00

* 5100<C100.CFFFF

* C0006: 00

4、完成以上工作后,基本完成了各个系统数据块与 U7 的定位。详细情况见下表。

内容地址	U7 逻辑地址	U7 物理地址	映射的 RAM	容量
DOS 系统	\$ 8000-\$ A9FF	\$ 0000-\$ 29FF	\$ 1000-\$ 39FF	10.5K
内容 ROM	\$ C100-\$ CFFF	\$ 4100-\$ 4FFF	\$ 5100-\$ 5FFF	4K
系统程序	\$ D000-\$ FFFF	\$ 5000-\$ 7FFF	\$ 6000-\$ 8FFF	12K
空闲区域	任意设置映射	\$ 2A00-\$ 3FFF	\$ 3A00-\$ 4FFF	5.5K

(三)再修改:

再修改任务是完成对系统 BASIC 解释程序的修改工作,使在键入 DS 命令后,解释程序能自动把 DOS 移入 RAM 的 \$ 9600-\$ BFFF 中自定位,并设置好有关参数。

1、修改控制字,使键入 DS 后机器能鉴别执行。

* 5498: 44 D3

2、修改 Reset 程序,使开机后不引导磁盘。

3、修改搬移 DOS 程序。

5600-8D	03	C0	8D	05	C0	A9	00
5608-A8	85	FC	84	FE	A9	80	85
5610-FA	A9	96	85	FF	EA	EA	EA
5618-8D	03	C0	8D	05	C0	A9	20
5620-8D	B0	C0	B1	FC	8D	02	C0
5628-8D	04	C0	91	FE	C8	D0	E8
5630-EA	EA	EA	E6	FD	E6	FF	A9
5638-AA	F5	FD	D0	DB	EA	EA	EA
5640-8D	02	C0	8D	04	C0	EA	60

4、修改入口程序。

62DA-20	80	97	EA	20	18		
62E0-97	EA	EA	4C	03	E0	60	EA

至此,修改工作全部完成,-机内 RAM \$ 1000-\$ 8FFF 完全对应于新的 U7 物理地址 \$ 0000-\$ 7FFF 的内容。

(四)存盘待用:

为防止繁锁修改后的信息丢失,可将其存入磁盘待用。

* BSAVE DU HUA DOS, A \$ 1000, L \$ 8000

这样,你便保存起了内存 RAM \$ 1000-\$ 8FFF 副本。

(五)固化:

EPROM 的固化,可以根据手中现有的固化卡,采取各种手段来完成,限于各位读者手中持有的固化卡不尽相同,这里不再赘述。我们使用的是重庆九龙计算机公司生产的中华机用 SPRING-27A EPROM 读写卡,很快就完成了固化工作。

四、几点说明

(一)、为了保证学习机硬件的完整性,建议你保存好原来的 U7,这样一旦需要 LOGO 语言时,可以再换上使用。

(二)、为了固化 DOS,你可以自购一片 EPROM,选择 27256 或 27C256 均可,目前这类芯片零售价在 20-30 元。

(三)、固化前键入的程序和实施的步骤要格外谨慎,有时一个字节错误就可能使整个固化工作失败。所以工作要耐心细致。反复核查无误后再进入固化阶段。

(四)、不要力图反复拷入和擦除 EPROM 中的内容。开始时,我们不太了解 EPROM 的寿命,经过一段时间后才发现,它的寿命一般在 10 次左右,反复的擦除写入或写入不当,就有可能使 R PROM 寿终正寝了。

功率谱估计的 C 语言实现

华中理工大学 张红庆

一、引言

随机信号的功率谱反映它的频率成分以及各成分的相对强弱,能从频域上提示信号的节律,因此用数字信号处理技术来估计随机信号的功率谱是统计信号处理的基本手段之一。

计算机高级语言中, C 语言简洁,灵活,具有丰富的数据类型,齐全的图形功能,而且可移植性强。笔者以 Turbo C2.0 为语言工具,编制了功率谱分析的实用程序。

二、功率谱估计的韦尔奇法

韦尔奇 Welch 法即平滑周期法的平均法。

信号序列 $x(n)$ 的长度为 N , 将其分成长度均为 L 的 K 段, 这些区段互相重叠 $L/2$ 个样点, 则第 i 段序列为: 式(1)

为减少信号序列分段时所引起的截段影响 (吉布斯现象), 将每段序列 $X(n)$ 乘以窗函数 $Wd(n)$ 进行加权, 有式(2)

$Wd(n)$ 可为 L 个点的矩形窗、汉明窗、哈宁窗等等。

然后对各加权段进行 M 点 ($M \geq L$) 的快速付里叶变换 (FFT), 即式(3)

得到各段修正的平均周期图式为 (4)

求平均产生归一化角频率 $w = 2\pi k / M$ ($0 < K < M-1$) 处的功率谱估计值式(5)

三、程序实现

程序流程图见图 1。

程序由 PM.C 和 ZZZ.C 两个程序组成。PM.C 中 FFT (采用时间抽选法) 和逆 FFT 变换函数及主函数, 主函数开始确定 FFT 的长度 M (必须是 2 的整数幂)、窗型、窗长 L 、数据采样数目 N (N 能被 $L/2$ 整除)、采样频率 FS , 然后计算均值, 减去均值后进行功率谱估计。

ZZZ.C 中有 mainwindow 屏幕主窗口形成函数及描绘曲线函数 gline, 主函数调用 gline 画出功率谱对数谱图及自协方差。

四、实例

假设模拟采样信号 $X(n) = \cos(2\pi n / 10)$, 取参数: $M = 128, L = 64, N = 256, FS = 10000\text{Hz}$, 将序列分为 7 段, 加用汉明窗。

程序运行后打印出来的对数功率谱图及自协方差见

图 2。

附图及程序清单:

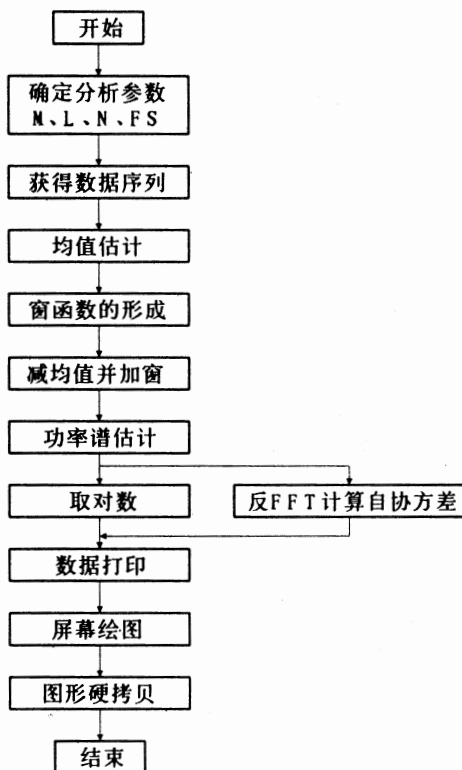
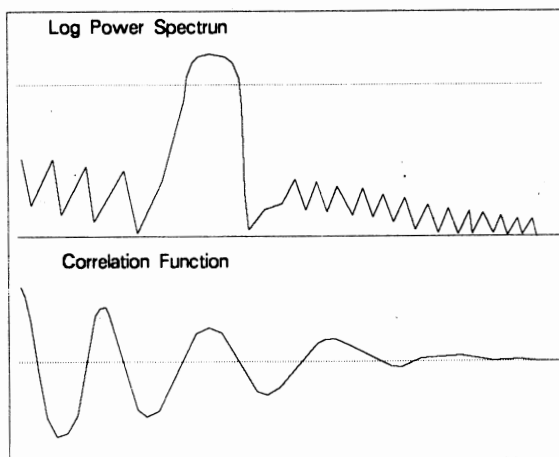


图 1 程序流程图



```

/* 程序PM.C */
#include <math.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <io.h>
#include <fcntl.h>
#include <dos.h>
#define PI 3.1415926536

float y[129],yy[129],y1[129],g[257];
typedef struct {
    float re;
    float im;} cplex;

/*复数加法运算函数*/
cplex add(cplex z1,cplex z2)
{
    cplex z;
    z.re=z1.re+z2.re;
    z.im=z1.im+z2.im;
    return(z);
}

/*复数减法运算函数*/
cplex sub(cplex z1,cplex z2)
{
    cplex z;
    z.re=z1.re-z2.re;
    z.im=z1.im-z2.im;
    return(z);
}

/*复数乘法运算函数*/
cplex mul(cplex z1,cplex z2)
{
    cplex z;
    z.re=(z1.re*z2.re)-(z1.im*z2.im);
    z.im=(z1.re*z2.im)+(z1.im*z2.re);
    return(z);
}

/*复数取模*/
float ccabs(cplex z)
{
    return(sqrt(z.re*z.re+z.im*z.im));
}

/*复数相角*/
float zang(cplex z)
{
    return(atan(z.im/z.re)*180/PI);
}

/*时间抽选法快速付里叶变换*/
void FFT(cplex x[],int m)
{
    cplex t,u,w;
    int n,nv2,nm1,j,k,le,le1,ip,l,i;
    nv2=ldexp(1,m);

```

```

    nv2=nv2/2;
    nm1=nv2-1;
    j=1;
    for(i=1;i<=nm1;i++)
    { if(i<j)
        { t=x[j];
          x[j]=x[i];
          x[i]=t; }
        k=nv2;
        while(k<j)
        { j=j-k;
          k=k/2; }
        j=j+k;
    }
    for(l=1;l<=m;l++)
    {
        le=pow(2,l);
        le1=le/2;
        u.re=1.0; u.im=0.0;
        w.re=cos(PI/le1); w.im=(-sin(PI/le1));
        for(j=1;j<=le1;j++)
        {
            for(i=j;i<=n;i=i+le)
            { ip=i+le1;
              t=mul(x[ip],u);
              x[ip]=sub(x[i],t);
              x[i]=add(x[i],t);
            }
            u=mul(u,w);
        }
    }
    for(i=1;i<=n;i++)
    { y[i]=ccabs(x[i]);
      yr[i]=x[i].re; }
}

/*反付里叶变换*/
void IFFT(cplex x[],int m)
{
    int n,i;
    n=ldexp(1,m);
    for(i=1;i<=n;i++)
    x[i].im=-x[i].im;
    FFT(x,m);
    for(i=1;i<=n;i++)
    yr[i]=yr[i]/n;
}

/*均值计算*/
float mean(float x[],int n)
{
    int i;
    float xx;
    xx=0;
    for(i=1;i<=n;i++)
    xx=xx+x[i];

```

```

xx=xx/n;
return(xx);
}

/*主函数*/
main()
{
int M,L,NN,K,i,FS,j,n;
unsigned char *title="x(i)=cos(2*PI*i/10) ";
unsigned char *title1="Log Power Spectrum";
unsigned char *title2="Correlation Function";
float U,Xmean,x[257],x1[257],w[129];
cplex xx[129];
M=128; L=64; NN=256; FS=10000;
for(n=1;n<=NN;n++)
x[n]=cos(2*PI*(n-1)/10);
Xmean=mean(x,NN);
for(n=1;n<=NN;n++)
x[n]=x[n]-Xmean; /*减去均值*/
for(n=1;n<=L;n++) /*生成窗函数*/
w[n]=0.54-0.46*cos(2*PI*(n-1)/(L-1));
U=0;
for(n=1;n<=L;n++)
U=U+w[n]*w[n];
K=(NN-L/2)/(L/2);
for(n=1;n<=M;n++)
yy[n]=0;
for(i=0;i<=K-1;i++)
{
for(n=1;n<=L;n++)
{ xx[n].re=x[n+i*L/2]*w[n];
xx[n].im=0.0; }
for(n=L+1;n<=M;n++)
{ xx[n].re=0;
xx[n].im=0.0; }
FFT(xx,7);
for(n=1;n<=M;n++)
yy[n]=yy[n]+y[n]*y[n]; }
for(n=1;n<=M;n++)
yy[n]=yy[n]/(K*U);
for(n=1;n<=M;n++)
{ xx[n].re=yy[n];
xx[n].im=0; }
for(n=1;n<=M;n++)
yy[n]=20*log10(yy[n]);
printf("Sample Function: x(i)=cos(2*PI*i/10) ");
printf("\n");
printf("Xmean=%8.4f",Xmean);
printf("\n");
printf("M=%d NP=%d L=%d SAMPLING FREQUENCY=%d\n",
M,NN,L,FS);
printf("\n");
printf("\n");
printf("LOG Power Spectrum");

```

```

printf("\n");
printf(" FREQ DB FREQ DB\n",
FREQ DB FREQ DB);
printf("\n");
for(i=0;i<=16;i++)
{ printf("\n");
for(j=1;j<=4;j++)
{ printf(" %8.2f", (float)(i*4+j-1)*FS/M);
printf(" %8.3f",yy[i*4+j]); }
}
printf("\n");
printf("\n");
printf("Correlation Function");
printf("\n");
printf(" LAG CORR LAG CORR\n",
LAG CORR LAG CORR);
for(i=0;i<=16;i++)
{ printf("\n");
for(j=1;j<=4;j++)
{ printf(" %8d", (i*4+j-1));
printf(" %8.3f",yr[i*4+j]); }
}
Initialize();
MainWindow();
gline(x1,0,title);
getch();
IFFT(xx,7);
MainWindow();
gline(yy,0,title1);
gline(yr,1,title2);
}

/* 程序ZZZ.C */
#define ESC 0x1b
/* Define the escape key */
#include <graphics.h>
#include <alloc.h>
#include <io.h>

char *LineStyle[] = {
"SolidLn","DottedLn","CenterLn","DashedLn","UserBitLn"
};
/* extern unsigned char xx[256]; */
int GraphDriver;
/* The Graphics device driver */
int GraphMode;
/* The Graphics mode value */
double AspectRatio;
/* Aspect ratio of a pixel on the screen*/
int MaxX,MaxY;
/* The maximum resolution of the screen */
int ErrorCode;
/* Reports any graphics errors */

```

```

struct palettetype palette;
/* Used to read palette info */

void DrawBorder(void);
void Pause(void);

void Initialize(void)
{
    int xasp, yasp;
/* Used to read the aspect ratio*/

    GraphDriver = DETECT;
/* Request auto-detection */
    initgraph( &GraphDriver, &GraphMode, "" );
    ErrorCode = graphresult();
/* Read result of initialization*/
    if( ErrorCode != grOk ){
/* Error occurred during init */
        printf(" Graphics System Error: %s\n",
            grapherrormsg( ErrorCode ) );
        exit( 1 );
    }
    MaxX = getmaxx();
    MaxY = getmaxy();
/* Read size of screen    int    GraphDriver;
/* The Graphics device driver */
}

float Mmax(float x[],int n)
{
    int i;
    float max;
    max=0.0;
    for(i=1;i<=n;i++)
        { if(x[i]>max)
            max=x[i];
        }
    return(max);
}

float Mmin(float x[],int n)
{
    int i;
    float min;
    min=1000.0;
    for(i=1;i<=n;i++)
        { if(x[i]<min)
            min=x[i];
        }
    return(min);
}

gline(float xx[],int n,unsigned char *title)
{ int i;

```

```

float max,min;
/* clearviewport(1,n*MaxY/2+1,*/
/*MaxX-1,n*MaxY/2+MaxY/2-1); */
max=Mmax(xx,64)-Mmin(xx,64);
min=Mmin(xx,64);

    if(n==0)
        { outtextxy(30,15,title);
          for(i=1;i<64;i++)
              { setlinestyle(2,0,1);
                line(8*i,(int) MaxY/2-(MaxY/3*(0-min)/max),
                    8*(i+1),(int) MaxY/2-(MaxY/3*(0-min)/max));
                setlinestyle(SOLID-LINE,0,1);
                line(8*i,(int) MaxY/2-(MaxY/3*(xx[i]-min)/max),
                    8*(i+1),(int) MaxY/2-(MaxY/3*(xx[i+1]-min)/max)
                );
              }
          }
    else
        { outtextxy(30,MaxY/2+15,title);
          for(i=1;i<64;i++)
              { setlinestyle(2,0,1);
                line(8*i,(int) MaxY-(MaxY/3*(0-min)/max),
                    8*(i+1),(int) MaxY-(MaxY/3*(0-min)/max));
                setlinestyle(SOLID-LINE,0,1);
                line(8*i,(int) MaxY-(MaxY/3*(xx[i]-min)/max),
                    8*(i+1),MaxY-(MaxY/3*(xx[i+1]-min)/max));
              }
          }
}

void MainWindow()
{
    int height;
    cleardevice();
/* Clear graphics screen */
    setviewport( 0, 0, MaxX, MaxY, 1 );
/* Open port to full screen*/
    DrawBorder();
    line(0,MaxY/2,MaxX,MaxY/2);
}

void DrawBorder(void)
{
    struct viewporttype vp;
    setlinestyle( SOLID-LINE, 0, NORM-WIDTH );
    getviewsettings( &vp );
    rectangle( 0, 0, vp.right-vp.left, vp.bottom-vp.top );
}

```

参考文献:

- [1] 江太辉 数字信号处理,华中理工大学出版社
- [2] 李桂青 微机 C 语言及应用,气象出版社

用 dBASE III 评估学生成绩

昆明第三中学物理组 魏明逊

在学期结束,教师对各科成绩进行分析是一件繁杂的工作,而且一般仅限于排名次、计算平均分。要做到不同学科之间的比较就很困难。我们利用 dBASE III 在管理上的优点,结合教育统计学编制了一套软件,大大地减轻了教师的负担,而且对学生的成绩分析更加科学合理准确无误。

一、教育统计学的基本内容

为了达到科学分析学生成绩的目的,我们除了对成绩进行排序外,还求平均分、标准差、标准分。计算公式如下:

$$\text{平均分: } \bar{X} = \frac{\sum_{i=1}^N X_i}{N}$$

$$\text{标准差: } S_x = \sqrt{\frac{\sum_{i=1}^N (X_i - \bar{X})^2}{N}}$$

$$\text{标准分: } Z_i = \frac{X_i - \bar{X}}{S_x}$$

二、利用 dBASE III 编制程序

首先我们要建立三个数据库: XSCJ3.DBF, XSCJG32.DBF, XSCJG33.DBF。

1. xscj3.DBF 数据库

建立名次,学号,姓名,以及各科的相应字段,由教师输入学号、姓名、各科成绩。

2. XSCJG32.DBF 数据库

.USE XSCJG32

.LIST

RECORD 分类 语文 数学 外语 物理
化学 生物

1. 平均分

2. = 100 分

3. 90 = < 100 分

4. 80 = < 90 分

5. 70 = < 80 分

6. 60 = < 70 分

7. 不及格

8. 50 = < 60 分

9. 40 = < 50 分

10. 30 = < 40 分

11. 30 以下

3. XSCJG33.DBF 和 XSCJG32.DBF 基本一样,只要把 XSCJG32 中的记录:平均分改为标准差就行。

XSCJG32.DBF 反映的是平均分和不同分数段的人数,而 XSCJ33 反映的是标准差和所占的百分比。

4. 教师只要录入学号、姓名、各科成绩,其余由计算机完成。我们以六科为例, XSCJG31.DBF 为成绩库, XSCJG34.DBF 为标准分库,由计算机自动建立。

本程序在 IBM PC / XT, AT, 386 机上调试通过,效果良好。程序中的过程文件 PR1.PRG 即打印程序,采用《电脑》杂志 1992 年第一期“一个自动输出表格的多功能 DBASE 程序”。

源程序清单:

```
set talk off
set procedure to prt
clear
clear all
d=dtoc(date())
use xscj3
replace all 总分 with 语文+物理+数学+英语+化学+生物
replace all 平均分 with 总分/6
sort on 总分/d, 数学/d, 语文/d TO XSCJG31
copy stru exte to aa
USE
USE XSCJG31
go hott
nc=recno()
NUM=0
DO WHILE NUM<NC
NUM=NUM+1
GO NUM
REPLACE 名次 with num
enddo
copy to xscjg34
clear all
sele 5
use xscjg34
index on 名次 to ixscjg34
y1=""
Y2=""
sele 1
use xscjg33
index on 分类 to ixscjg33
sele 2
use xscjg32
index on 分类 to ixscjg32
```

```

SELE 4
use xscjG31
index on 名次 to ixscjg31
go bott
nc=recno()
go top
SELE 3
use aa
coun to n2
n=2
do while n<n2
n=n+1
sele 3
GO N
x=field-name
if x="语文" .OR. X="数学" .OR. X="英语" .OR. X="物理";
.OR. X="化学" .OR. X="生物" .OR. X="平均分"
Y1="Y"
ENDIF
sele 4
IF Y1="Y"
COUNT TO A1 FOR &X=100
COUNT TO A2 FOR &X>=90 .AND. &X<100
COUNT TO A3 FOR &X>=80 .AND. &X<90
COUNT TO A4 FOR &X>=70 .AND. &X<80
COUNT TO A5 FOR &X>=60 .AND. &X<70
COUNT TO A6 FOR &X<60
COUNT TO A7 FOR &X>=50 .AND. &X<60
COUNT TO A8 FOR &X>=40 .AND. &X<50
COUNT TO A9 FOR &X>=30 .AND. &X<40
COUNT TO A10 FOR &X<30
average &x to b
SELE 2
close index
GO 1
REPLACE &X WITH B
GO 2
REPLACE &X WITH A1
GO 3
REPLACE &X WITH A2
GO 4
REPLACE &X WITH A3
GO 5
REPLACE &X WITH A4
GO 6
REPLACE &X WITH A5
GO 7
REPLACE &X WITH A6
GO 8
REPLACE &X WITH A7
GO 9
REPLACE &X WITH A8
GO 10
REPLACE &X WITH A9
GO 11
REPLACE &X WITH A10
SELE 1

```

```

USE XSCJG33 INDEX IXSCJG33
SELE 2
USE XSCJG32 INDEX IXSCJG32
SELE 1
UPDAT ON 分类 FROM XSCJG32 REPLACE &X WITH;
B->&X,&x WITH &X/NC*100
sele 5
num=0
tl=0
t=0
do while num<nc
num=num+1
go num
t=(&x-b)*(&x-b)
tl=tl+t
enddo
sx=SQRT(tl/nc)
sele 1
go 1
replace &x with sx
sele 5
use xscjg34 index ixscjg34
sele 4
use xscjg31 index ixscjg31
sele 5
updat on 名次 from xscjg31 replace &x with D->&x,&x;
with (&x-b)/sx
ENDIF
Y1=""
ENDDO
CLEAR ALL
USE XSCJG34
COPY TO XSCJ
ZAP
USE
USE XSCJ
COPY TO XSCJG34 FIELDS 名次,学号,姓名,语文,数学,;
英语,物理,化学,生物,平均分
USE
CLEAR ALL
CLEAR
@ 8,20 SAY "数据库名: XSCJG31"
DO PR1
CLEAR
@ 8,20 SAY "数据库名: XSCJG32"
DO PR1
CLEAR
@ 8,20 SAY "数据库名: XSCJG33"
DO PR1
clear
@ 8,20 SAY "数据库名: XSCJG34"
DO PR1
close procedure
RETURN

```

参考资料:

- [1] 姜溥仁、陈子正主编《中学物理教学法》
- [2] 徐民材主编《汉字 dBASE III 实用教程》

邬家炜

4.

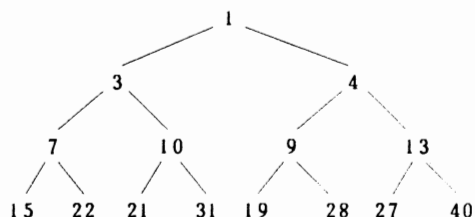
- 39

```

10 S=0:R=0
20 FOR I=1 TO 6:S=S+I:NEXT I
30 FOR J=1 TO 6
40 FOR K=1 TO 6:IF J=1 THEN 180
50 FOR L=1 TO 6:IF K=1 OR K=J THEN 170
60 FOR M=1 TO 6:IF L=I OR L=K THEN 160
70 FOR N=1 TO 6:IF M=I OR M=J OR M=K OR M=L THEN 150
80 N=S-I-J-K-L-M
90 IF I+J+K<>I+N+M OR I+J+K<>K+L+M OR I+N+M<>K+L+M THEN 150
100 R=R+1
110 PRINT " ";I
120 PRINT " ";R;" ";J;" ";N
125 PRINT " ";K;" ";L;" ";M
130 PRINT:PRINT".....":PRINT
150 NEXT M
160 NEXT L
170 NEXT K
180 NEXT J
190 NEXT I
200 END

```

第五题：本题不难解的方法也多。下面我们根据题意，用二叉树来表示 M 中的数如下图



这里树根的值 1，对树中任一节点 X，其左儿子的值为 $2X+1$ ，右儿子的值为 $3X+1$ 。不难理解，如果该树用一维数组 A 存储，那么对树中任一节点 A(I)，其左儿子为 A(2*I)，右儿子为 A(2*I+1)，根据题意要求产生 M 的前 100 个最小数，故该树应有足够的深度，不妨取深度 $K=9$ ，这时有 $2^K-1=511$ 个节点，即生成 A(511) 中所有节点的值，再排序，最后输出前 100 个。下面给出一参考解答。

```

10 DIM A(511):A(1)=1
20 FOR I=1 TO 225
30 Y=2*I:A(Y)=2*A(I)+1:GOSUB 80
40 Y=2*I+1:A(Y)=3*A(I)+1:GOSUB 80
50 NEXT I
60 FOR I=1 TO 100:PRINT A(I):NEXT I
70 PRINT:END
80 T=A(Y):J=Y-1
90 IF A(J)<T THEN A(J+1)=T:RETURN
100 A(J+1)=A(J):J=J-1:GOTO 90

```

第六题：这是一道模拟球的弹跳运动的题目。按题意，我们知道小球抛出的速度水平分量是 $V_x = V_0 \cos 45^\circ = 7.071$ （其中 V_0 表示初速度），它不因弹跳而改变，因此水平方向的运动是匀速的。在垂直方向，小球的运动可由方程：

$$Y = V_0 \sin 45^\circ \cdot t - \frac{1}{2} g t^2$$

来描述。在球着地之前，Y 始终取正值。着地时 $y=0$ 。判别球的着地时间，只要看 y 何时为 0。超过这个时间， $y<0$ 着地以后球在垂直方向按

$$y = V_0 \sin 45^\circ \times 0.9t - \frac{1}{2} g t^2$$

运动，其中 t 是以着地的时间作为起点计算的。

以上是描述问题的主要算法。

下面是供大家参考的一个解答。

```

10 FOR I=1 TO 60
20 PRINT "-";
30 NEXT I
40 PRINT TAB(61);"Y"
50 LET N=1
60 LET V1=10*SIN(45*3.14159/180)
70 LET M=1
80 LET T1=0
90 FOR T=0 TO 10 STEP .01
100 LET T1=T1+.01
110 LET Y=V1*T-.5*9.8*T*T
120 IF Y=0 THEN 170
130 IF M=1 THEN 150
140 GOTO 160
150 PRINT TAB(0);"1";TAB(1+20*Y);" "
160 GOTO 220
170 LET T=0
180 LET N=N+1
190 LET V1=V1*.9
200 IF N>4 THEN 270
210 GOTO 110
220 LET M=M+1
230 IF M=10 THEN 250
240 GOTO 260
250 LET M=1
260 NEXT T
270 PRINT TAB(1);"X"
280 END

```

第七题：这是一个难度较大的题。按题意不妨把图中的 7 个区域划分如下图：

我们先对区域 1 着颜色“1”，并找出与区域 1 不相邻的区域，它们可以用集合 B 来描述： $B=\{2, 4, 5, 6, 7\}$

选择区域 2，同样着颜色“1”，注意与区域 2 相邻的区域 3，4，8 将其中数字与 B 集合中相同的从集合 B 中除去，因为它们既然与区域 2 相邻，就不可能着颜

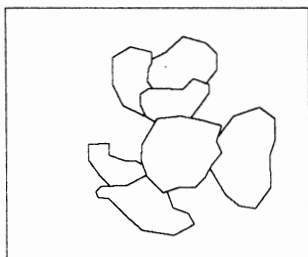
色“1”。着色后的区域2也从集合B中除去，剩下的是 $B' = \{5, 6, 7\}$

对区域5着色“1”，注意它与区域3, 6, 7, 8相邻，在B'中除去6, 7及5。B'变成一个空集，着第一种颜色的区域到此为止。

对于区域3着色“2”，与区域3不相邻的区域是集合B中的6, 7，对区域6也着以颜色“2”，并从集合B中除去6。注意到与区域6相邻的是区域5, 7, 8，在集合B中除去7后，B变成了空集。

区域4可以着色“3”，与区域3不相邻的区域7同样着色“3”。

最后对图的底着色“4”。



下面是一个参考的解答。

```
10 DATA 1,3,8,0
20 DATA 2,3,4,8,0
30 DATA 3,1,2,4,5,8,0
40 DATA 4,2,3,8,0
50 DATA 5,3,6,7,8,0
60 DATA 6,5,7,8,0
70 DATA 7,5,6,8,0
80 DATA 8,1,2,3,4,5,6,7,0
90 INPUT N
100 DIM A(N,N),B(N)
110 FOR I=1 TO N
120 FOR J=1 TO N
130 LET A(I,J)=0
140 NEXT J
150 NEXT I
160 FOR I=1 TO N
170 FOR J=0 TO N
180 READ A(I,J)
190 IF A(I,J)=0 GOTO 210
200 NEXT J
210 NEXT I
220 LET C=1
230 FOR I=1 TO N
240 IF A(I,0)<0 GOTO 260
250 GOTO 280
260 NEXT I
270 GOTO 530
280 FOR K=1 TO N
```

```
290 LET B(K)=0
300 NEXT K
310 LET E=1
320 FOR J=1 TO N
330 IF I=J GOTO 430
340 LET M=1
350 FOR K=1 TO N
360 IF A(I,K)<>J GOTO 380
370 LET M=0
380 NEXT K
390 IF M=0 GOTO 430
400 IF A(J,0)<0 GOTO 430
410 LET B(E)=J
420 LET E=E+1
430 NEXT J
440 LET A(I,0)=-C
450 LET G=1
460 LET P=B(G)
470 IF P=0 GOTO 510
480 FOR K=1 TO N
490 IF A(K,0)=P GOTO 580
500 NEXT K
510 LET C=C+1
520 GOTO 230
530 PRINT "AREA","COLOUR"
540 FOR L=1 TO N
550 PRINT L,"(",-A(L,0);")"
560 NEXT L
570 END
580 LET A(K,0)=-C
590 LET B(G)=0
600 FOR J=1 TO N
610 IF A(P,J)=0 GOTO 670
620 FOR D=2 TO E
630 IF A(P,J)<>B(D) GOTO 650
640 LET B(D)=0
650 NEXT D
660 NEXT J
670 LET G=G+1
680 IF G>E GOTO 510
690 LET P=B(G)
700 IF P=0 GOTO 670
710 GOTO 480
RUN
? 8
AREA COLOUR
1 (1)
2 (1)
3 (2)
4 (3)
5 (1)
6 (2)
7 (3)
8 (4)
```

煮酒论英雄

——大型游戏“三国演义”全略

佛山 卫 易

如今随着电脑的普及化,许多家庭已拥有了电脑。那么在平时的学习过程中,如果有几套神奇的游戏作品,在闲暇空余时玩上个把小时,那也是一种极好的娱乐方式。

本人以前极少玩游戏,因为模拟游戏实在是打得有些无聊,况且其说明全是英语或者日语,让人看了头疼。而台湾“智冠”公司的不少中文版的游戏却可以说是有了质的飞跃。

从英文版的“水浒传”到“三国志”,从汉语版“楚汉相争”到“三国演义”和“春秋争霸”,一套比一套精彩,本人最喜欢其中的“三国演义”,现将一些游戏的体会和心得列举在此,供大家参考。

游戏容量达三兆半,需要硬盘。AT机,1兆内存,支持单显(最好是VGA),如果有魔音卡那玩起来,完全就是一种享受。键入SG倒车后,系统会提示你选择是游戏卡或硬盘;你的显示方式;是否有魔音卡。等待一会,屏幕显示标题等,然后是一幅极为壮观的图画,火红的赤壁仿佛是当年刘、孙、曹决战的地方,战船破浪摆开战场,接着屏幕显示出那一首千古绝唱的词作,“滚滚长江东逝水,浪花淘尽英雄,……青山依旧在,几度夕阳红。”屏幕将显示主要人物的形象。并有刘、孙、曹及关、张的大型图画。

按空格键进入游戏后,有六大功能选择,其中第六功能为退出游戏;第五功能为音乐欣赏,供有魔音卡的电脑用户使用;第三和第四功能为字体选择,可选择楷书和隶书汉字;由于游戏庞大,有时需要几天的功夫才能统一中国,所以在游戏过程中提供游戏存储,第二功能便是调用储存的游戏,最多储存六个游戏;第一功能便是开始新游戏。

游戏将中国领土分为四十二郡,消灭所有对手,统一中国者为胜者。

新游戏有六个年代供你选择,第一个年代是董卓专权时代,董卓兵精粮足,拥有十万大军,刘备兵不过三百,将不过关、张。孙坚也是苦心经营,即使是袁绍、曹操等人也难以立足,笔者也只有试过选择袁绍、刘焉苦战三十个春秋才获胜;第二年代为官渡之战前夕,袁绍势力虽大,但曹操也拥兵自立,谋臣良多,刘备拥有两郡,稍微有些资本,孙策也在江南虎视眈眈,加上袁术、吕布、刘表等人,都有机会一展鸿图;第三个年代为官渡之战之后,袁绍虽败,但仍拥有实力可以和曹操一战,刘备寄人

篱下,困守新野,四面强敌包围,形势险恶。而孙权拥江东数郡,随时准备重复历史上的赤壁之战;第四年代为赤壁之战后,刘备实力大增,虎视益州,意图吞并刘璋,孙权也是雄心勃勃,志在中原,曹操更是以绝对的优势藐视孙、刘;第五个时代为刘备占领益州,三国鼎立之势已成,关、张、赵、马、黄名将如虎,良臣如云的时候。纵然是曹、孙也夜梦难眠;第六年代为关羽大意失荆州之后,刘备疆土减少,但名将还多,依然可以再酬大志,你可千万不要意气用事,让陆逊有机会火烧连营三百里。此时有姜维、邓艾、钟会等人出现,又是一片崭新的气象。

你可以选择其中的一个年代,每一年代都有许多君主供你选择,如刘、孙、曹之外还有马腾、袁术、吕布、张鲁等十数个人物。如果你愿意,也可以将自己的名字输入电脑,选一块土地加入这场战争。时代越是靠后,所可以选择的人物就越少。这游戏也可以几个人同时玩,但很不实际。

电脑会提示你选择难度,1-10或11-20,其区别在于前十种两军交战,主帅被消灭或被俘虏,则全军败,后十种余部则还可以继续战斗。二者都是数字越大,难度越大。建议你尽量玩难些,会增加许多乐趣。

开始游戏后,作为君主的你拥有同一般人不同的权力,所以要尽量使用。而且如此多的人才各有所长,你一定要充分利用才行。每个游戏中的人物都会在谋略、战斗力、魅力等上有一定的点数,100点为最高,其中谋略最高的是诸葛亮一百点,其次是庞统、司马懿、周瑜等。战斗力最高为吕布一百点,其次是张飞、关羽、赵云(99点)等人,魅力最高为刘备(100点)。

作为君主,要注意观察自己拥有各郡的土地值、洪水率、自己的人望等,这些因素往往对你的成败至关重要。

你有九大功能可以选择,每一大功能还有许多小功能,先作以下简单叙述。

功能0为状态,一般游戏停留在此功能上来显示当前郡的土地值、洪水率、人民的忠诚度,以及君主的人望和这郡的太守、将领数、军队数量、粮草金银数量等。

功能1为查询,可以供你随时查询你所拥有的郡的状态(某个或者全部),以及这些郡的地理地图和你所拥有的物品(宝刀、骏马、美女、兵书等)。同样你也可以查询敌方的状态,但这将占用你一个月的时间。

功能2为军事，你可以任意地按照你的需要来调动你的军队，也可以挥动马鞭来发动战争，并且可以将战争物资用最快的速度运输到任何属于你的郡。

功能3为兵士，你可以训练士兵，当他们的训练达到100点时，战斗力就最强。你可以征兵，扩充你的军队，补充你的兵源。当然你还需要购买大量的武器。

功能4为内政，你可以促进土地开发，使这郡尽量生产你所急需的粮草金银。你也要尽量降低这块地的洪水率，减少天灾带来的巨大损失。你还可以用多余的金银建筑营寨来增强你的防卫能力。

功能5为商业，当你的郡收入实在太低，而你又急需金银时，你不妨尝试着做一些小买卖，乘粮价高的时候卖出，乘粮价低的时候买进，反复多次，你就会有一些收入。如果你粮食充足自然还可以放仓济民，提高百姓对你的忠诚度。

功能6为人事，你可以寻找在此隐居的名士，登用他们，赏赐他们金银。同样可以因为他们的不称职而撤职。

功能7为君主，是你的特权。你首先要命名一个谋略超群的军师，这是你胜利的开始，成功的保障。而将一个郡交给一个人管理，这个人的魅力、谋略、战斗力也要让你费点脑筋。将你所拥有的物品来赏赐给你的将领，不但可以提高他们对你的忠诚，而且可以提高他们各方面的能力。你还可以将某郡交给手下自治，这样可以大大刺激那里的生产力。如果发现对方的将领有人不够忠诚，不妨尝试一下招降，如果那将领还是太守，那你就可以不费一兵一卒获得一个郡，当然，这还取决于你这君主的能力和人望。

功能8为谋略，是君主和军师的特权，你可以驱虎吞狼，使敌人自相残杀；你也可以远交近攻，利诱你的敌人和你联手去攻打另一敌人；伪书使疑，可以离间敌方君主和臣子的关系；此外还有策反人民；联合出兵可以联合自己两个郡的军队攻打对方一个郡，纵然给对方识破，那也等于发动一场20个兵团参战的大战，异常激烈。

功能9为其它功能，例如存储、结束、调用音乐功能等。这里值得一提的是，不管你的电脑有没有魔音卡，只要它有只可以出声的喇叭，那么打开语音功能，调解适当的深度，你将可以使游戏中所有的人物说话。比如当赵云率领自己郡的大军进攻张辽防守的郡时，他会说：“张辽，汝多行不义，吾将伐之”张辽也不会客气，“赵云匹夫，安敢欺吾”有意思吧？

以下再介绍一下本人玩这套游戏的一些体会和心得。

一、虽然你一定是一个出色的君主，但请注意你的职责在于指挥，运筹帷幄之中，决胜千里之外。所以不妨将你的郡交给你所信任的将领自治。我可以保证，他们治理能力。

二、作为君主，应该找一个相对安全的地方驻扎，如果建立起后方就更好。

三、一金可以征一个兵，一将最多可带的士兵人数由将领的能力而有一千五百人至三千人不等，君主最多可以

带兵五千人，一个郡最少需要百姓三千人，可以驻扎五十员大将，十五万人马（这已是极庞大的数字）。

四、注意地图上各郡的位置，选择战略要地占领，尽量分割敌人的疆土，使他们彼此间互相不能联系。但敌人也会在这些地方和你反复争夺。

五、金银粮草是决定战争胜负的另一个极为关键的因素，你所作的一些都需要金银，征兵需要，战争需要，来登用人才都要。如果你的将领谋略高，那拥有三千金银可以抵一万精兵。如果你没后方，或你的金银太少，那就将对方的防线撕开去敌后抢夺。（真的好象古代的战事！）

六、每一年开始会有大量人才出现，四月有一场洪水，七月丰收之时，往往就是战争的开始，尤其当你还弱时。

七、洪水率一定要降低，否则一场洪水会无情地使你一个郡的军队、金银粮草损失一半以上。

八、孙子兵法说的伐谋在这游戏中可以体现出来，你一定要任命一个谋略超群的军师，让他为你来出谋划策。运用功能8中的远交近攻，使两股敌人火拼，而你在一旁坐收渔翁之利，何乐而不为呢？而且战争结束后你俘虏的不但是你所要进攻的敌酋。许多前来助攻而被俘的将领也被你所擒。

九、在战争的过程中，请尽量运用你所派出将领的谋略，在天时、地理合适的情况下出奇制胜，一定能一举成功。例如对方军团驻扎在树林间，又是风起的日子，如果你的将领谋略有80点以上，且比对方高一些，用火攻一把火就可以将对方烧得片甲不留。当然这也要用钱（600金）。同样在靠水的地方，趁大雨时用水攻（500金）。最经济实惠的谋略还是围攻，集中自己几个兵团围攻对方一个兵团，战斗力顿时倍增，对方的主帅战斗力最是强，提议你先集中所有的弓箭来射击，然后各个击破。但你千万不要低估你的对手，也许失败的是你。

十、假如对方的郡是孤立的，又或者对方的地理是一环形山脉，仅有几个出口，而我军实力又远不及对方强，那可以烧毁对方的粮草辎重，当对方粮食用尽时，你就可轻易获胜。注意：尽量全歼你的敌人，否则对方见大势不妙，会溜走的。等他们整顿阵容反扑过来，那就功亏一篑了。

十一、如果所有的条件对你都不利，那就只有集中你最大的力量和资产，攻击对方最薄弱的环节，放火烧粮。还不行，那只有脚底抹油，逃之夭夭了。留得青山在，不怕没柴烧，这道理相信你一定能懂。

这是一个对你的智慧、忍耐力、观察分析力进行挑战的有些游戏。相信你一定可以发挥你在军事、政治上的潜力，一尝你一统中原霸主天下的梦想。而且此游戏变化多端，实在让人爱不释手。不妨请你试下第一、第三年代的刘备，太棒了。

正是：秦失其鹿，天下共逐；

谁主沉浮，君付一笑。

第二代病毒的产生及其对策

福建炼油厂 何善华 王东姝

【摘要】本文首先对病毒史作了一个概括性的回顾,接着首次正式提出了第二代病毒的概念,并详细阐述有关第二代病毒的特征及反病毒技术发展的现状,最后向读者简单介绍了一个设计思想独特、可靠有效的能保护用户资源、安全带毒运行的防抗病毒系统 *Millennium AntiVirus V3.0*。

一、电脑病毒史的回顾

自 1987 年美国发现第一起蠕虫病毒破坏电脑系统资源以来,病毒以一个极其迅猛的速度滋生并广泛地传播到世界各地。病毒的危害是如此之大,侵袭范围如此之广,以致世界各主要国家都纷纷制订下有关电脑安全的法律或管理条例,以期最大限度地遏制来势汹汹的病毒大军。而众多的软件公司更是不遗余力地开发研制各种消毒软件,也有少数几家公司已开发了一些据称能预报警的软(硬)件。尽管如此,病毒的兴盛看来丝毫没有衰微之势。因为反病毒技术的兴起,反过来也促进了病毒技术(如果可以称之为技术的话)的革故鼎新,大有道高一尺、魔高一丈之感。

回顾病毒的变迁就可证实上述所言不虚。早期的病毒,也就是第一代病毒,最初多以所谓引导型病毒的面目出现,后来,经过病毒与反病毒技术的不断对抗及升级,逐步出现了文件型(或称外壳型)病毒以及混合型病毒。它们的发展是逐步的、多侧面的,甚至是高技巧和高水平的。

一般地,引导型病毒可分为三大类:

1、覆盖式主引导型病毒

此类病毒总是覆盖硬盘主引导扇区,它们的传播方法多为由软盘启动系统时带入,传播的媒介或途径是软盘,一般性的特征是带毒软盘启动时硬盘指示灯亮不止一次(而 MS-DOS 仅只操作一次),引导系统时间增加,扇区常见的信息消失,取而代之的是不成文字信息的机器码。

2、搬迁式主引导型病毒

搬迁式主引导型病毒也具有覆盖式主引导型病毒的许多特征,但因为此类病毒将主引导扇区移走,取代它的是病毒体本身,加之它们处处对用户设防,当用户要检查主引导扇区时病毒总是将原来的信息提供给用户,并

拒绝主引导扇区的写操作,所以,在病毒控制之下,要发现及消除此类病毒对一般用户来说有一定的难度。

3、操作系统引导扇区型病毒

操作系统引导型病毒不去影响硬盘的主引导扇区,只是简单地将操作系统的原引导记录移走,把病毒体的一部或全部写到引导扇区。它们都具有引导型病毒的共同特征。

早期的引导型病毒自身的缺陷十分明显,即使有许多加密、伪装技巧等依附于上,也总被有经验的用户或适当的消毒软件清除。

文件型病毒:

文件型病毒的形式更是多种多样,它们的发展最能在某种程度上体现软件技术的一些新动向、新发展。如果我们能以不同的角度看待病毒这个事物,不难从中获取一些正面的、有益的东西。

最简单的文件型病毒,是只能依附于 .COM 文件的一种,这种病毒感染传播完全依赖于 .COM 文件,其运作及占有控制权亦依赖于 .COM 软件的运行。此种病毒的基本特征是, .COM 文件自动有限或无限地增长,某些关于盘操作的时间明显增加,较糟糕的一种情形是盘操作或系统运行速度变慢到了不能容忍的地步。

稍为复杂一点的文件型病毒,则能感染 MS-DOS 可执行文件的两种最基本形式 .EXE 文件和 .COM 文件,这种病毒显然亦具上述特征,但消除的难度增大,一般用户不能胜任。

能感染 .COM、.EXE、.OVR、.OVL、.OVY、.OBJ、.SYS 的文件型病毒是颇具威胁性的。它们不但具有了文件型病毒的共同特点,更具有了嵌入程序体的功能,从而为扫描及消除此类病毒增加了相当的难度,这正是消毒软件所不能及之所在。

许多安全专家预期的源程序嵌入式病毒至今未有发现,这种病毒在单用户 MS-DOS 下估计短时间内不太可能产生,不仅是因为难度大、语言版本多,更主要的是因为 MS-DOS 是通过指令级与用户发生联系的。故要将代码嵌入各种语言源程序内,病毒体的代码无疑要占用较多的内存,容易被发现并清除。不过,我们不能完全排除这种病毒产生的可能性,因为要研制这样的病毒在理论及实现上都是可行的。

混合型病毒:

到目前为止,混合型病毒乃具最大危害及隐蔽性的一种病毒。它们具有引导型病毒及文件型病毒的各自形式与特征,又能交叉感染、互相作用。只要其中任何一种形式被加载,另一种形式的病毒体也就可能同时被引入。此类病毒之所以难于清除,原因是多方面的,但其中有一点往往被人忽略或不愿正视,即用户在清除了这类病毒的其中一种形式之后,常常自信大功告成,根本还没想到当你清除了病毒的一种外在形式,另一种形式的病毒体正伺机而动、图谋卷土重来呢。

二、第二代病毒的特征种种

第一代病毒的技术发展已日臻成熟,虽然反病毒技术也在相互对抗中得到提高,但并不真正对层出不穷的病毒构成什么威胁。相反,病毒的制作因此更其精巧、更其狡猾,终于导致了第二代病毒的诞生。

简单地说,不具备或很少具备,或即使具备了第一代病毒特征但在病毒的控制之下却无法用通常方法检查的病毒都是第二代病毒。这种病毒的又一个显著特征是不着痕迹,并体现在它们的传播与表现上。在外观上,它们也有第一代病毒的几种基本形式,但可能以混合型病毒为主流。

下面,我们仅就第二代混合型病毒为例,讨论其一些具体的常见的特征,而不准备在理论上做过多的探讨。当然,引导型及文件型病毒的讨论也已在剖析的过程中包含了。

早期的病毒,经验用户都可以凭外观感知其存在。比如:

1、软盘启动时硬盘灯亮了不少一次

现在,病毒可以在操作系统操作硬盘时再行感染硬盘,其余时候潜伏忍耐,所以在软盘启动时硬盘灯亮亦止一次。虽然其操作硬盘绝对超过一次,但由于硬盘读写速度高,靠外观感知显然是不行的。

2、INT 13h 中断被修改

这种单纯的观察,很早以前就失效了。因为

MSDOS 自版本 3.0 起为了适应日益发展的存贮介质,已对该中断做了扩充与完善,而且软驱数量的增多使各厂家也纷纷推出自己的设备驱动程序。

3、内存量被减少

INT 12h 可获取的内存量(在 0: 413h 处,一字)被减少,而病毒就安置于因此占有的内存块处。

DOS 对内存容量的获取是一次性的,以后若直接对 0:413h 处进行任何操作,它都将置之不理,此举显然存在严重的缺陷。因为病毒可以很容易地判断系统是否已引入了 MS-DOS,若已引入,则将 0:413h 处的原内存量恢复,而 MS-DOS 却全然不觉,它不会将程序代码加载到自己早已获取的内存高端以外的位置。

4、工具软件可发现 MS-DOS 可用内存量与系统内存量的不一致性

有些用户,过分依赖于象 PCTOOLS 一类的工具软件查询系统配置,去了解病毒的存在。然而,PCTOOLS 之类的工具软件只是简单地从系统数据区或某些特定的内存单元取析数据,因而它们的局限性也是不言而喻的。在第二代病毒出现以前,用 DEBUG 直接查看数据段偏移 2 处一字的内容是更简单有效的办法,它从中不仅可知内存量的变化,还可直接知道病毒寄生的段域。

遗憾的是,这种方法并不特别可靠。因为众多的 CC-DOS 在增加了卸出功能后,存在很多的问题。主要的两个问题一是将用户安装在较 CC-DOS 更高层的驻留代码“擅自”一并卸出,另一问题就是可能卸得不干不净,以至 MS-DOS 在内存高端处留下“疵块”(这里操作系统自身的缺陷也显而易见,但本文对上述问题不准备发表过多的议论)。这样,用户会误将内存高端的不当占用,视为病毒代码的寄存。

然而,第二代病毒可以让你根本无法用以上介绍的方法探知其存在。问题还在 MS-DOS 本身。为什么呢?原来,MS-DOS 自身保留有 COMMAND 外壳的第一副本,其 PSP 保留了它原先获取的以及初始设置的信息,当 MS-DOS 为用户安装 PSP 时,有些信息只是简单地从自己保留的 PSP 里复制出来、反馈回去。病毒正是利用这个特点,将用户 PSP 有关内存量的内容加以伪装。不过,这种技术较为复杂,搞不好易造成死机。

5、INT 21h 有了与平常不同的殺值

第二代病毒将不再单纯以 INT 21h 作为攻击目标,它们将转而攻击其它诸如 INT 25h,INT 26h,

INT 27h 以及 INT 40h 等从前未被注意的几个中断。即使攻击 INT 21h, 也可能只搜索其中的诸如 CALL [wreg], CMP AH, breg 之类的指令, 然后将其改向 (发 INT 指令或段间调用、跳转指令), 而不直接改写 INT 21h 中断及其向量。这种技术是目前已知的被病毒采用的最复杂的一种, 其实现的难度最大。

6. 盘操作速度变慢

这种判断方法是幼稚的。殊不知, 病毒完全可能使你的盘操作速度明显“加快”呢。其原因很简单, 病毒只需放弃用户在用户数据区内的盘操作或只操作其中一部分, 即可达到加速的效果。

7. 文件长度增加

早期的某些病毒, 已能对文件长度增加这个缺陷作一些伪装, 比如在用查找匹配首文件及其它匹配文件时, 病毒会将文件的原来长度给出, 从而迷惑用户, 但不很彻底。第二代病毒的伪装将更完善, 即使用指针移至文件末部及 FCB 方法查找匹配文件都将被病毒迷惑, 用户已不能放心且方便地调用 MS-DOS 功能调用了。

8. 引导记录不正常

一般的资料都会提醒读者, 观察引导记录的正常信息是否消失或被替换, 等等。现在, 引导记录仍然被修改, 但修改量很少, 并且代码是正常的, 不认真追踪不易察觉。

9. 其余特征

第二代病毒最大的特点是不露痕迹。体现在传播上时, 是读写盘或操作文件的速度没有直观上的变慢, 体现在发作方式上时, 则不会有口号或即使有也不公开, 它们一般不会给显示器一个明显的干扰, 但可能会蚕食或象“黑洞”那样, 不知不觉地吞食一些显示字符。而在攻击串、并行口设备时, 也不是简单地取消, 更可能的是输出中有“疵点”, 若表现在文件输出上, 这样的破坏将更具危害。假如你在编制程序, 由于病毒的作用 (插入“疵点”), 你必须花数以倍计于平时的时间去调试你的代码, 并且由于总不能解决问题, 工作将不能继续进行, 除非你发现了病毒的存在并加以清除。在汉字操作系统, 各种输出可能屏蔽汉字, 但我们往往会认为汉字系统出了问题, 或者软件不适于在汉字系统下运行, 也可能认为诸如打印机之类的外设出了故障。

由于反病毒技术的滞后和落后, 制作病毒的动机各异 (有些病毒是源于反复制等保护性措施, 见下文), 因而病毒设计的队伍日见庞大, 有些人甚至将其视为通

向 MS-DOS 的捷径, 因此它们的特征也必然是形形色色的, 本文就不再一一罗列了。

病毒的蔓延, 其中重要的一个原因是非法复制软件的现象大量存在, 许多实用软件为了防止及惩罚侵权行为, 都可能在程序体内植入病毒代码。所以, 为了免遭病毒危害, 应加强道德及法制观念, 不非法复制、使用未经授权的软件, 不要因小失大, 也不要过分自信。因为植于高级语言编译的可执行代码内的病毒很难被发现和检测。一般地, 购置软件应注意是否有用户手册等必要的资料, 最好要弄清该软件的出版者是否提供有售后服务及升级服务。

三、反病毒技术的现状

自从病毒的蔓延及其危害被人们认识以来, 反病毒技术的发展也是很迅速的。然而, 无庸讳言, 到象 Millennium AntiVirus 这样优秀的软件出现之前, 反病毒技术在总体上是逊于病毒的, 在这场长期的对抗中处于下风地位。滞后的扫描、消毒怎么可能与不断翻新花样的病毒一试锋芒呢? 只要对病毒稍事修改, 或仅修改病毒的标志, 就能逃过消毒软件的检查, 任何一个病毒, 都可以产生几百种甚至几千种的变异。因此, 这种反病毒功能虽然在一定的阶段、一定的程度上起了一些积极的作用, 但目前从其总体效果甚微来看, 实在不值得称道和发扬。

反病毒硬卡的产生正是为了弥补消极扫描软件的严重不足, 在当前市场上也可谓百花齐放、各显神通。究竟硬卡反病毒有多大的作用? 我们只需认真分析一下不难得出结论。我们知道, 市面上各种汉卡、反毒卡都只是软件的固化形式, 一些是基于特殊应用的特殊考虑, 而大部分的反毒卡纯粹是为了防止非法复制或迎合用户对硬件的迷信才硬化的, 并不会比软件更安全、更有效; 相反, 病毒程序只需三、五条指令就能置硬卡于死地, 硬卡之脆弱不言而喻。况且, 目前的硬卡都缺少智能, 总是在不适当的时候不厌其烦地向用户作假报告, 增添麻烦不说, 许多工作需要自动连贯地进行, 但在硬卡的干预下, 不回答它们的问题, 用户的工作就别想再进行了, 自动连贯工作从何谈起。

最有希望彻底阻止引导型病毒的方法之一是完善 MS-DOS 本身。事实上, 增加不到十条指令就可以做到。然而 MS-DOS 不去正视自身的缺陷, 据说将来要推出的 MS-DOS 6.0 将增加反病毒命令, 果真如此, 可谓舍本逐末了。

而较困难的是文件型病毒的检测, 为了不失开放

性, MS-DOS 对自身的保护甚少, 而且对用户应用软件的管理也很单纯 (并且薄弱, 虽然只是单用户操作系统, 然而其内存管理存在不尽人意的地方已不少了), 这给文件型病毒以大大可趁之机。

反病毒技术的发展最终回归于依靠软件或软件与硬件的配合, 已是大势所趋, 也是反病毒技术逐渐走向成熟的开始。当前最优秀的一些软件已能在很大程度上自动识别许多第一代病毒, 虽然它们尚不能预警第二代病毒, 又可能被针对性的病毒击破, 但比从前那种被动脆弱又不可靠的反病毒技术来说已是前进了一大步, 不可同日而语。

在反病毒的过程中, 有一些令人不安的因素至今还没有完全排除。某些有危害性的反病毒软件或硬卡也在市场上露面。那种给磁盘种“疫苗”、给文件种“牛痘”的做法是荒唐可笑的, 然而某些直接具有破坏性的软件硬卡竟也堂而皇之地招摇过市, 其原因主要是设计者对 MS-DOS 及病毒的机理了解不足, 对用户的资源使用情况闭门造车、自以为是。

看来, 反病毒技术的突破必须建立在对病毒机理的透彻认识、对 MS-DOS 的深刻了解以及具有丰富的实际经验之上, 三者缺一不可。Millennium Software Central (以下简称 MSC) 推出的 AntiVirus V3.0 是众多反病毒软件中的佼佼者, 本文愿向读者慎重推荐之。

四、Millennium AntiVirus V3.0 防抗病毒系统

Millennium AntiVirus (以下简称 AntiVirus) 的研制显然是建立在对病毒的共性 (不管是第一代的还是第二代的, 还是未来的病毒) 和对 MS-DOS 及其许多重要数据的分析消化之上。她被设计得十分精巧, 虽然没有那种华丽的外表, 然而却极其有效。她对自身和 DOS 以及所有可能受病毒攻击的单元、数据都层层设防, 病毒根本无法攻击受 AntiVirus 保护的电脑系统。为了预防万一, MSC 又专门提供了 AntiVirus V3.0 的多种变异版本, 这样, 要攻破 AntiVirus 的防线概率为零。此外, MSC 已公开声明, 保证质量的同时, 保证将来有升级服务与售后服务。因此, 用户尽可放心使用 Millennium AntiVirus。

首先, AntiVirus 的安装十分简单方便, 根本不需用户做过多的干预。用户只需用购得的 AntiVirus 源件启动电脑系统, 除非发现你的电脑系统已经异常, 否则 AntiVirus 立即与原有资源融为一体。今后, 任何病毒都难逃脱 AntiVirus 的控制。

其次, AntiVirus 本身设计得十分安全可靠, 不象

市面上某些反病毒软件那样具有危险的副作用。AntiVirus 的策略是获取用户正当的资源并加以保护、利用, 而不修改用户任何有用的数据、单元。

第三, AntiVirus 不但防抗病毒独具匠心, 用于保护用户资源也是卓有成效, AntiVirus 的这种功能已超出了反病毒的范畴。受其保护的电脑系统, 非授权用户无法对其进行操作、访问, 甚至用 DOS 的 FDISK 破坏也不能奏效。另外, 如上文所述, 当用户使用了代码设计得不好的程序时, 可能造成系统资源的浪费或导致系统崩溃, 而 AntiVirus 竟也可以发现这样的问题并加以修复。

第四, 任何引导型病毒的侵袭都将被立即清除, 任何带有文件型病毒的程序都可照常运行, 而不对系统构成威胁, 换句话说, 用户可以安全带毒运行而不用担心运行效率会降低。

另外, 设计思路再巧妙、再怪僻的病毒也不能企望逃脱 AntiVirus 的追踪、封杀。事实上, AntiVirus 研制长达四年, 经历过包括第二代病毒在内的针对性测试, 仍稳如泰山。由于 AntiVirus 具有未来预见性, 病毒设计者很难巴望制造一种能躲过 AntiVirus 侦查、控制的新病毒, 除非该病毒不是运行于 MS-DOS 之上。

AntiVirus 还有很多特点, 比如不会不当干预用户日常工作, 重实质不重外在, 使用户几乎感觉不到她的存在, 如果没有病毒侵袭的话。AntiVirus 还加强了对 MS-DOS 的保护, 这也是在所有其它反病毒系统中所未曾见的。

总之, AntiVirus 的诞生, 给反病毒技术输入了新鲜血液, 带来了新的希望。也许, 病毒与反病毒的对抗将因此进入一个新的时期。

本文参考了国内外多种报刊有关介绍病毒的文章及众多的软件、硬卡, 由于篇幅所限, 不能一一列于参考文献之中, 在此特予申明。另外, 对于在本地了解 Millennium AntiVirus 有困难的读者, 欢迎与敝下联系。

参考文献:

- [1] Millennium Software Central: Millennium AntiVirus V3.0 user reference, 1993 年
- [2] 李沐荪编译: MS-DOS 操作系统高等教程, 北京, 科海总公司培训中心, 1987 年 9 月
- [3] 张载鸿编: 局部网操作系统 DOS 高级技术分析, 第一版, 北京, 国防工业出版社, 1988 年 12 月

单片微机与模糊控制讲座

第三讲 模糊控制理论

广东工学院 余永权

模糊控制就是采用模糊控制器对被控对象进行自动控制的控制方法。在模糊控制中，由计算机执行由条件模糊语句描述的控制规律，并把对象控制在给定的性能指标处。在这一讲中，介绍模糊控制理论中的模糊化、精确化和模糊控制规则问题。

3.1 模糊控制的基本原理

模糊控制是建立在人的客观经验的基础上的，而控制过程和一个专业人员的操作过程类同。模糊控制在本质上就是一种智能型控制。

一个专业人员对生产过程的控制基本分成三个步骤：

- 1、用眼、耳等器官，观察生产过程的输出状态，并把输出精确量转成模糊信息存于脑中。例如，人们观察到生产过程输出的温度达 800°C ，就形成“温度相当高”这种模糊信息。
- 2、根据取得的信息，并按自己的经验进行判别，从而确定应如何对生产过程的控制信息进行调整。
- 3、在确定出进行控制的模糊信息之后，在具体执行时则采用相应的精确量。例如，在确定“把阀门关小一些”这一模糊控制信息之后，实际执行时是把阀门旋转一个精确的角度。

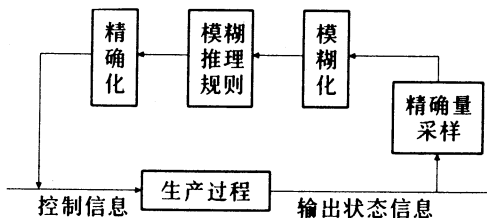


图 3-1

很明显，专业人员进行人工控制生产过程时，是执行对生产过程的信息进行模糊化，按经验进行模糊推理，把推理结果得到模糊信息精确化去控制生产过程。

要用计算机仿照专业人员那样进行生产过程控制，就必须使计算机执行输入信息模糊化，按输入信息进行模糊推理产生模糊控制信息，把模糊控制信息精确化后去控制对象。也就是说建立一个模糊控制系统。

很容易知道，一个模糊控制系统如图 3-1 所示。在图中，模糊推理是一条条经验规则，有时也简称为模糊规则。

在实际系统中，采样，模糊化，模糊推理精确化都由计算机执行。通常，能执行采样、模糊化，模糊推理及精确化这几种功能的控制装置称为模糊控制器。因此，在模糊控制系统中，包括有模糊控制器，反馈环节和被控对象，如图 3-2 所示。通常在模糊控制系统中，为了提高系统的控制水平，不但把输出量 Y 和给定值 R 进行比较产生偏差 e ，而且还对偏差求变化率 Δe ，并且把 e 和 Δe 送去模糊化再执行推理。虽然，在一些要求简单的系统，只用偏差 e 去作为推理的输入量就可以了。但是，在有一定性能要求的系统中，都采用 e 和 Δe 作为推理的输入量。因为这样做，在计算机模糊控制系统中，并没有增加任何硬件，只是增加了计算量而已。

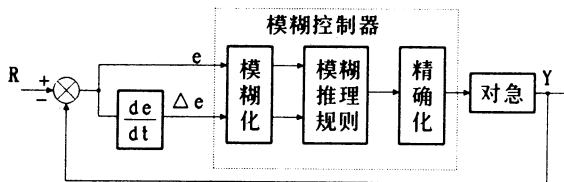


图 3-2

3.2 输入量的模糊化

输入量的模糊化是模糊控制中特有的一种处理过程。在模糊控制中，模糊控制器的输入量是偏差 e 和偏差变化率 Δe ；所以，模糊化是对 e 和 Δe 而言的。

在系统中，给定信号也就是期望系统输出所达到的指标。把系统的实际输出值和给定信号进行比较就可得

到偏差 e 。通常给定信号由 R 表示。系统输出信号用 Y 表示, 故而偏差 e 为:

$$e = R - Y \quad (3-1)$$

无论是给定信号 R 还是系统输出信号 Y , 它们都是精确值, 当然, 偏差 e 也就会是精确值。模糊化的处理过程包括两个步骤:

1. 定义模糊量

这个步骤根据实际需要确定有关模糊量及其意义。例如, 在一个温度控制系统, 给定信号是 100°C 。则系统输出的温度和给定信号之间的差值是偏差 e 。

当给定值 R 大于系统输出值 Y 时, 偏差 e 为正; 否则为 0 或负。根据偏差的正负及大小可以定义偏差的 7 种模糊量, 即正大 (PB), 正中 (PM), 正小 (PS), 零 (Z), 负小 (NS), 负中 (NM), 负大 (NB; 而且各个模糊量的意义如图 3-3 所示。

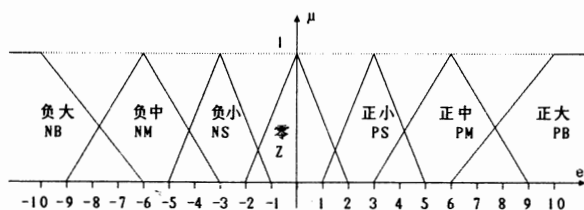


图 3-3 e 模糊量的定义

在图中从正中 (PM) 这个模糊量为例, 它包括偏差 e 为 $+3 \sim +9$ 度的范围, 不过在这个范围中值不同, 则对正中 (PM) 这个模糊量的隶属度不同。显然有:

$$\mu_{\text{PM}}(3) = 0, \quad \mu_{\text{PM}}(4) = 0.33, \quad \mu_{\text{PM}}(5) = 0.66$$

$$\mu_{\text{PM}}(6) = 1, \quad \mu_{\text{PM}}(7) = 0.66, \quad \mu_{\text{PM}}(8) = 0.33$$

$$\mu_{\text{PM}}(9) = 0$$

其余模糊量的定义范围及意义也同样可以从图 3-4 中得出。当然, 模糊量也可以用表格定义。模糊量正中 (PM) 用表格可表示如下:

偏差 e	3	4	5	6	7	8	9
正中(PM)	0	0.33	0.66	1	0.66	0.33	0

在图 3-4 中所定义的模糊量只是一种人为的经验; 在实际中可以根据不同的经验和要求进行定义。

2. 把精确量转化为模糊量

这个步骤就是根据现行偏差 e 的精确值, 求出对应最合适的模糊量。例如, 当偏差值 e 为 -3 时, 则可以把其转化为模糊量负小 (NS)。有时, 精确值属于两个模糊量; 这时, 转化结果由精确值对模糊量的隶属度确

定。也就精确量应转化为隶属度高的模糊量。例如, 当偏差 e 为 4 时, 从图 3-3 看出; 它对正小 (PS) 的隶属度比对正中 (PM) 的隶属度要大, 因为有:

$$\mu_{\text{PS}}(4) = 0.5, \quad \mu_{\text{PM}}(4) = 0.33$$

故这时 $e=4$ 转化为模糊量正小 (PS)。

在模糊控制系统中, 定义模糊量是组成控制系统的先期工作之一; 控制系统运行时, 执行的是把精确量转化为模糊量这一操作; 这个操作通常是通过计算机查表完成的。

3.3 模糊控制规则及推理

模糊控制规则是专门的人员在对生产过程进行控制时经验的总结。这些经验用一条一条控制规律表示, 故称控制规则。这些控制规则反映了人对控制过程的判别和控制决定, 即人对事物变化的推理。

按照人所总结的模糊推理规则就可形成一种抽象的模糊控制器, 从而取代人执行自动控制。

由于控制规则有多条, 每一条都可以从其输入和输出间求出相应的输入输出关系, 对一个控制系统来说, 则要求总的输入输出关系。有了总的输入输出关系之后, 一旦有某种输入, 就可以求出相应的输出。在控制器中, 输入的是偏差 e , 输出的是控制量 u 。

由于模糊控制器输出的是模糊量, 故控制量 u 也是模糊量。所以, 对 u 也应象对偏差 e 那样进行类似模糊定义。如果系统需用到偏差变化率 Δe 时, 也应进行类似的模糊定义。

对于一个电炉恒温系统, 人们得出如下一些经验:

- 1、如果偏差为正大, 则控制量为负大;
- 2、如果偏差为正中, 则控制量为负中;
- 3、如果偏差为正小, 则控制量为负小;
- 4、如果偏差为零, 则控制量为零;
- 5、如果偏差为负小, 则控制量为正小;
- 6、如果偏差为负中, 则控制量为正中;
- 7、如果偏差为负大, 则控制量为正大;

一般而言, 上述经验用模糊条件语句来表示, 并且

有:

IF $e = \text{PB}$ THEN $u = \text{NB}$;

IF $e = \text{PM}$ THEN $u = \text{NM}$;

IF $e = \text{PS}$ THEN $u = \text{NS}$;

IF $e = 0$ THEN $u = 0$;

IF $e = \text{NS}$ THEN $u = \text{PS}$;

IF $e = \text{NM}$ THEN $u = \text{PM}$;

IF $e = \text{NB}$ THEN $u = \text{PB}$;

在上面规则中, 要注意; 控制量为 0, 不是真正取

值为 0，这里是模糊量“0”，它的值可以是某种电压；比如控制量“0”是取 170V 左右的值。并且有控制量的模糊定义如图 3-4 所示。

从第一条控制规则可以求出输入输出关系 R_1 ，并且有：

$$R_1 = PBe \times NBu \quad (3-2)$$

在求取关系 R_1 时，是以 PBe 为列向量， NBu 为行向量执行模糊矩阵的求交操作的。显然，根据 c 和 u 的定义有：

$$PBe = (0, 0, \dots, 0, 0.25, 0.5, 0.75, 1)^T$$

$$NBu = (1, 0.75, 0.5, 0.25, 0, \dots, 0, 0)$$

注意， PBe 和 NBu 都是含有 21 个元素的向量，也即是 PBe 前 17 个元素为 0， NBu 的后 17 个元素为 0。这一点分别从图 3-3，图 3-4 可以看出。从而关系 R_1 是一个 21×21 的模糊关系矩阵，它的元素由列向量 PBe 和行向量 NBu 对应元素求交得出，即：

$$R_1 = [r_{ij}]_{21 \times 21} \quad i = 1, 2, \dots, 21 \quad j = 1, 2, \dots, 21 \quad (3-3)$$

对 7 条控制规则进行处理，则可得 $R_1 \sim R_7$ 这 7 个关系矩阵。系统的总关系为 R ：

$$R = \bigcup_{i=1}^7 R_i \quad (3-4)$$

得出系统总关系 R 之后，对于任何输入偏差 c ，都可以求出输出控制量 u ：

$$u = c \circ R \quad (3-5)$$

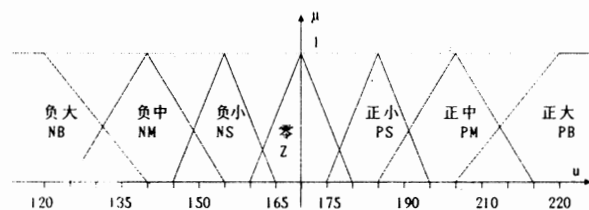


图 3-4 控制量 u 的模糊定义

这样，也就完成了控制中的推理过程。控制规则的处理，关键就是通过这些规则求出可供推理总关系矩阵 R 。

3.4 输出量的精确化

从上面输出控制量 u 的求取过程可知道，得到的控制量 u 是一个模糊量。在实际控制过程中，加给被控对象的控制信号只能是一个精确值，不可能是一个含有多个不同隶属度的值组成的模糊量。因此，通过模糊控制器求得的模糊量 u 必须处理或精确量才能控制对象。把模糊量处理成精确量的过程也称精确化。精确化一般有三种方法。

1. 最大隶属度法

因模糊控制器输出的控制量 u 是一个模糊量，故它必有下列形式

$$\tilde{u} = \mu_u(x_1) / x_1 + \mu_u(x_2) / x_2 + \dots + \mu_u(x_n) / x_n$$

$$\text{或 } \tilde{u} = \sum_{i=1}^n \mu(x_i) / x_i \quad (3-6)$$

如果存在最大隶属度 u_{\max} ，并且

$$\mu_{\max} = \mu_u(x_m) > \mu_u(x_1), \dots, \mu_u(x_{m-1}), \mu_u(x_{m+1}), \dots, \mu_u(x_n)$$

则取 $\mu_u(x_m)$ 对应的元素 x_m 作为精确控制量 C 故有：

$$C = x_m \quad (3-7)$$

当模糊控制量 u 中含有多个最大隶属度时，则其对应元素相加后的平均值为精确控制量 C 。例如有：

$$\mu_{\max} = \mu_u(x_j) = \mu_u(x_k) = \mu_u(x_l)$$

则精确控制量为：

$$C = (x_j + x_k + x_l) / 3 \quad (3-8)$$

2. 中位数法

中位数法利用模糊控制量 u 的隶属函数曲线和元素坐标所围成的面积并把这个面积分成两个相等的部分，分界线对应的元素为精确控制量 C 。

如果存在

$$\sum_{i=1}^n \mu_u(x_i) = \sum_{i=n+1}^{n+m} \mu_u(x_i) \quad (3-9)$$

则取精确控制量 C ：

$$C = x_n \quad (3-10)$$

3. 加权平均法

在加权平均法中，一般取隶属度为加权系数。在这种方法中精确控制量 C 由下式给出：

$$C = \frac{\sum_{i=1}^n \mu_u(x_i) \cdot x_i}{\sum_{i=1}^n \mu_u(x_i)} \quad (3-11)$$

通常，把模糊控制量 u 进行精确化的过程，就是在模糊控制量 u 判决出相应的最有代表性的精确量的过程，故而这个过程也称模糊判决，或模糊决策。

数据库卡片式查询通用程序

总后三五四一厂微机室

莫胜强

在利用 Foxbase 编制应用程序时,通常都要设置数据查询部分。但因所查阅的数据库结构各有不同,而需相应编制多种屏幕格式程序,特别是采用卡片式屏幕格式时,编程繁琐,重复劳动严重。针对这一问题,笔者编制了一个数据库分页卡片式查询通用程序。此程序可将任意数据库的记录整理为分页式卡片输出到屏幕上,整齐、美观,极为实用。可省去编写查询程序的烦恼,提高编程效率,又可获得满意的视觉效果。

几点说明:

一、程序使用极为简单,只须给出数据库文件名即可。决定整个程序的参数只有几个: ksh-显示开始行; ksl-显示开始列; mpxs-每屏显示行数。若需改变显示位置,可仅调整这三个参数来满足不同需要。

二、程序在取得用户所给数据库后,用 copy stru exte <filename> 的方法将库结构拷贝给变量待用。单独使用本程序,用此法十分简单方便。但如果要将此程序加到其他系统中,应考虑固定 tcxk.dbf 库的结构,并增加两个字段;其一登记字段的中文提示;其二作为是否显示的标志,并在无用字段不显,从而使显示效果简洁、合理,更好地服务于其他系统程序。

三、应在所使用的 Foxbase 中的 Config.fx 文件文件加入 "mvcount=700" 一条,以保证所需内存变量空间。如果库文件字段数超过 100,所需内存变量空间则更大。

四、程序在 AST386/33、长城系列微机在 Foxbase 上运行通过。

```
set talk off
set safety off
set color to 7/0
clear
@ 0,0 to 24,79
@ 20,1 to 20,78
@ 21,13 say "【数据库查询】 ★解放军第三五四一工厂★"
@ 22,1 to 22,78
```

```
@ 23,3 say "↑,↓,PgUp,PgDn - 定位, Esc-退出"
ksl=4          && 开始列
ksh=3          && 开始行
jxz=80-ksl-4   && 每屏最大宽度
if int((jxz-ksl)/2) <> (jxz-ksl)/2
    jxz=jxz-1
endif
mpxs=8         && 每屏行数
sjkm=""
@ 10,20 say "请输入数据库名: " get sjkm
read
sjkm=trim(sjkm)
if .not. file(sjkm+".dbf")
    clear
    return
endif
@ 23,36 say "★ 库名: "+sjkm
@ 23,53 say "★ 记录号: "
@ 23,68 say "★ 页号: "
sele 2
use & sjkm
fco=fcount()
copy structure extended to tcxk
sele 1
use tcxk
nu=1
@ ksh-1,ksl-2 clear to ksh+mpxs*2,jxz+2
do while nu<=fco.and..not.eof()
    nc=iif(nu>9,str(nu,2),str(nu,1))
    fname& nc=field-name
    ftype& nc=field-type
    flen& nc=field-len
    fsum& nc=flen& nc+len(trim(field-name))+4
    if int(fsum& nc/2) <> fsum& nc/2
        fsum& nc=fsum& nc+1
    endif
    fdec& nc=field-dec
    skip
    nu=nu+1
enddo
nu=2
zhh1=ksh
zbl1=ksl
page=0
do while nu<=fco
    nc=iif(nu>9,str(nu,2),str(nu,1))
    nc-1=iif(nu-1>9,str(nu-1,2),str(nu-1,1))
    zhh& nc=zhh& nc-1
    zbl& nc=zbl& nc-1+fsum& nc-1
    if zbl& nc+fsum& nc>jxz
        zbl& nc=ksl
```

```

    zbh& nc=zbh& nc-1+2
endif
if zbh& nc>ksh&mpxs+2-2
    zbh& nc=ksh
endif
if zbl& nc-1=ksh. and. zbh& nc-1=ksh
    page=page+1
    pg=iif(page>9, str(page, 2), str(page, 1))
    page& pg=nu-1
endif
nu=nu+1
enddo
sele 2
nu=1
page=1
do while .t.
    hx1=""
    n=1
    do while n<=(jxz-ksh-2)/2
        hx1=hx1+"—"
        n=n+1
    enddo
    hx1=hx1+"—"
    n=2
    do while n<=mpxs
        nc=iif(n>9, str(n, 2), str(n, 1))
        hx& nc="├"+substr(hx1, 3, jxz-ksh-2)+"┤"
        n=n+1
    enddo
    nc=iif(n>9, str(n, 2), str(n, 1))
    hx& nc="└"+substr(hx1, 3, jxz-ksh-2)+"┘"
    fcoc=iif(fco>9, str(fco, 2), str(fco, 1))
    hu=nu
    zbh0=0
    zbl0=0
    do while hu<=fco
        hc=iif(hu>9, str(hu, 2), str(hu, 1))
        hc-1=iif(hu-1>9, str(hu-1, 2), str(hu-1, 1))
        cc1=iif((zbh& hc-ksh)/2+1>9, str(
((zbh& hc-ksh)/2+1, 2), str((zbh& hc-ksh)/2+1, 1))
        cc2=iif((zbh& hc-ksh)/2+2>9, str(
((zbh& hc-ksh)/2+2, 2), str((zbh& hc-ksh)/2+2, 1))
        if zbl& hc<>ksh
            if substr(hx& cc1, zbl& hc-ksh+1, 2)=""
                hx& cc1=stuff(hx& cc1, zbl& hc-ksh+1, 2, "└")
            else
                hx& cc1=stuff(hx& cc1, zbl& hc-ksh+1, 2, "┘")
            endif
            hx& cc2=stuff(hx& cc2, zbl& hc-ksh+1, 2, "└")
        endif
        hu=hu+1
        hc=iif(hu>9, str(hu, 2), str(hu, 1))
        if hu>fco. or. (zbh& hc=ksh. and. zbl& hc=ksh)
            if hu>fco
                hx& cc2="└"+substr(hx& cc2, 3, jxz-ksh-2)+"┘"
            endif
        endif
        exit

```

```

    endif
enddo
n=1
do while n<=val(cc2)
    nc=str(n, 1)
    @ ksh+(n-1)*2-1, ksh say hx& nc
    if n<=val(cc1)
        @ ksh+(n-1)*2, jxz say " | "
    endif
    n=n+1
enddo
do while nu<=fco
    nc=iif(nu>9, str(nu, 2), str(nu, 1))
    ffff=fname& nc
    if ftype& nc="N"
        ffff=str(& ffff, flen& nc, fdec& nc)
    else
        ffff=& ffff
    endif
    ffff=trim(fname& nc)+": "+ffff
    @ zbh& nc, zbl& nc say " | "+ffff
    nu=nu+1
    nc=iif(nu>9, str(nu, 2), str(nu, 1))
    if nu>fco. or. (zbh& nc=ksh. and. zbl& nc=ksh)
        exit
    endif
enddo
@ 23,63 say str(recno(), 4)
@ 23,76 say str(page, 2)
do while .t.
    ink=inkey(0)
    do case
        case ink=27
            clear
            clear all
            close data
            return
        case INK=5
            if page=1
                if bof()
                    loop
                else
                    nu=1
                    page=1
                    skip -1
                    exit
                endif
            endif
            page=page-1
            pg=iif(page>9, str(page, 2), str(page, 1))
            nu=page& pg
            exit
        case ink=24
            if nu>=fco
                if eof()
                    loop
                else

```

```

        skip
        nu=1
        page=1
        exit
    endif
endif
page=page+1
pg=iif(page>9, str(page, 2), str(page, 1))
nu=page& pg
exit
case ink=18
if bof()
loop
endif
skip -1
nu=1
page=1
exit
case ink=3
if eof()
loop
endif
skip
nu=1
page=1
exit
case ink=1
if bof()
loop
else
go top
nu=1
page=1
exit
endif
case ink=6
if eof()
loop
else
go bott
nu=1
page=1
exit
endif
endcase
enddo
@ ksh-1, ksl-2 clear to ksh+mpxs*2, jxz+2
enddo

```

279



Turbo C在非中文系统下 显示和放大汉字的实现

华东工学院计算机系 邱素刚

在很多情况下，一些软件非常需要在不起动中文系统下能显示和放大汉字。笔者在对中文系统了解的基础上，用 Turbo C 写了显示和放大汉字的通用函数，简单而方便。今介绍如下：

汉字的显示通常是以点阵画在屏幕上的。每个汉字在字库中的点阵信息其起始地址与区位码有着唯一的对应关系，其公式如下：

$$\text{首址} = (\text{区码} - \text{汉字偏移区}) \times 94 + (\text{位码} - 1) \times 32$$

(其中偏移区据各字库有所不同)

而输入汉字文件的存贮一般都是用两个字节的汉字机内码，它们与区位码也有唯一对应的关系，其公式为：

$$\text{区码} = (\text{头一字节机内码} \& 7\text{FH}) - 32$$

$$\text{位码} = (\text{第二字节机内码} \& 7\text{FH}) - 32$$

(其中“&”表示逻辑与运算，7FH 是十六进制 7F) 这样，我们可以根据输入的汉字，推算出其点阵信息首址，读出点阵信息后，很容易用画点注显示和放大汉字。

对各函数作如下简单说明：

init(): 初始化屏幕为图形状态，打开汉字库。

long getcode (char[2]): 由机内码返回区位码。

get DZ (long, unsigned char [16] [2]): 计算汉字在字库中首址，读出 32 字节的点阵字码。

C: write (int x, int y, unsigned char bb[16] [2]) 坐标 (图形坐标) 处显示横向扩大 ax 倍，纵向扩大 ay 倍，颜色为 C 的点阵汉字 bb [16] [2] 的一个字。
cc_write (int x, int y, char msget, [] color): 在 (x, y) 处显示。

一汉字字符串 msgc, 颜色为 color 所定，不放大。done(): 关闭图状态。

main() 作为一个显示和放大汉字的例题，读者只要把源程序稍加修改，即可成为自己的通用显示和放大汉字的函数。

```
#include <stdio.h>
#include <string.h>
#include <graphics.h>
FILE *fpl;
void init();
long getcode(char [2]);
void getdz(long, unsigned char [ ] [ ]);
void c-write(int, int, unsigned char [ ] [ ], int, int, int);
void cc-write(int, int, char [ ], int);
void done();
main()
{ int i, j, k;
  unsigned char aa[5][16][2];
  long qw[5];
  extern FILE *fpl;
  char str[2];
  static char *bz="华东工学院";
  init();
  for(i=0, k=0; i<5; i++, k+=2)
  {
    str[0]=bz[k];
    str[1]=bz[k+1];
  }
  cc-write(240, 5, bz, 3);
  for(i=0; i<=4; i++)
    getdz(qw[i], aa[i]);
  for(k=1; k<=3; k++)
    for(i=0; i<=4, i++)
      c-write(20+i*(20+k*20), 20+20*(k-1)*(k-1), aa[i], k, k, k+1);
  cc-write(100, 180, "汉字显示放大程序华东工学院计算机系", 3);
  done();
  return 0;
}

void init()
{
  int gd=DETECT, gm, err;
  extern FILE *fpl;
  initgraph(&gd, &gm, "");
  err=graphresult();
  if( err !=grOk)
  { printf("Graphics error; %s\n", grapherrormsg(err));
    printf("Press any key to halt;\n");
    getch();
    exit(1);
  }
  sethbkcolor(1);
  if(! (fpl=fopen("\\cdoes\\cclib", "rb")))
  { printf("cannot open file\n");
    exit(1);
  }
}

long getcode(char str[2])
{
  return(((long) (((str[0]&0x7f)-32)*100+(str[1]&0x7f)-32)));
}

void getdz(long code, unsigned char bb[16][2])
```

```
{ int i, j;
  long dz;
  extern FILE *fpl;
  dz=(long) (((code/100)-9)*94+(code%100)-1)*32);
  fseek(fpl, dz, 0);
  for(i=0; i<16; i++)
    for(j=0; j<2; j++)
      bb[i][j]=fgetc(fpl);
}

void c-write(int x, int y, unsigned char ba[16][2], int a;
int ay, int c)
{
  int i, j, a;
  unsigned char ab[16];
  setfillstyle(1, c);
  for(i=0; i<16; i++)
  {
    for(j=0, j<16; j++)
      ab[j]=0;
    a=ba[i][0];
    for(j=0; j<8; j++)
    { if(a&1)
      ab[7-j]=1;
      a=a>>1;
    }
    a=ba[i][1];
    for(j=0; j<8, j++)
    { if(a&1)
      ab[15-j]=1;
      a>>1;
    }
    for(j=0; j<16; j++)
      if(ab[j])
        bar(ax*8+j+1, ay*i+y+2, ax*(j+1)+x, ay*(i+1)+y+1);
  }
}

void cc-write(int x, int y, char msge[], int color)
{
  int i, j, k;
  long code;
  char str[2];
  unsigned char data[16][2];
  for(i=0, k=0; i<strlen(msge)/2; i++, k+=2)
  { str[0]=msge[k];
    str[1]=msge[k+1];
    code=getcode(str);
    getdz(code, data);
    c-write(x+i*16, y, data, 1, 1, color);
  }
}

void done()
{ extern FILE *fpl;
  fclose(fpl);
  getch();
  setcolor(2);
  closegraph();
}
```

利用不可逆函数产生密码

四川省南充教育学院电教馆 罗永强

当前计算机管理中,在系统安全检查上常采用对照口令的方式,通常是在执行某种特定操作时要求用户输入口令,然后计算机根据用户输入的值查对权限表,判断操作是否合法,以确定计算机是否继续执行。计算机系统的失密常常是由于用户丢失口令,或存储在计算机中的权限表被破译。因而防止系统失密的关键有两点:一是用户口令的保管,二是防止非法用户获得权限表。对于前一种需要计算机用户妥善保存口令,不属于计算机系统工作范围,本文不作探讨;关键是后者,这也是计算机能否保密、系统是否安全的关键。

现在大多数采用 DBASE、FOXBASE 编制的管理系统,在使用口令时,先由用户输入口令,计算机对照系统密码,判断两值是否相等、操作是否合法。计算机对照口令只是单纯的关键值比较,对照的密码值存储在程序或数据库中,这对于熟悉计算机的人,常常可以轻而易举通过阅读程序或数据库获得系统密码,进而得到用户口令。虽然现在大多数用 DBASE、FOXBASE 等编制的程序可通过伪编译来防止程序代码被破译,但在一些工具软件如 DEBUG、PCTOOLS 面前也无任何秘密可言,因而除了采用常规的保护措施外,还要防止系统密码失窃,即是加强系统权限表的存储管理。

笔者从密码存储角度出发找到一种增强系统密码防窃能力的方法,使得即使通过工具软件查到存储的系统密码,也无法获得用户口令。

这种方法的思想是,不直接存储系统密码,将系统密码按一定加密算法计算后再存储在计算机中。假设系统密码是 KEYS,加密算法是 LOCK(),用户输入口令是 PASSW,首先将系统密码 KEYS(用户应输入的口令),用加密算法 LOCK()运算后得到一值 KEYS'存储在计算机中。计算机检查口令时,先由用户输入口令 PASSW,经加密算法 LOCK()运算后得到 PASSW',比较 KEYS' 和

PASSW',如果两值相等,则操作合法,否则操作非法拒绝执行。这样一来用户应输入的正确口令(系统密码)不直接存储在计算机中,用户口令不会从程序中被破译。

对于这种方法来说加密算法和存储的密码仍可以从程序中得到,如果加密算法选取不合理,就可能由存储的密码和加密算法逆推出用户口令,这样的系统仍不安全。但是如能找到一种恰当的加密算法,使得不能根据密码和加密算法逆推出用户口令,那么即使获得密码和加密算法,也不能推导出正确的用户口令。实际上数学中的不可逆函数能满足这种要求,由于从不可逆函数的函数值不能推导出其自变量值,因而即使知道了加密算法和密码,也得不到用户口令,防止了系统的失密。

利用现有程序设计语言构造不可逆函数是比较容易的,大多数程序设计语言都有 ASCII 码转换函数 ASC(),通过该函数可将字符型值转换成数值,只要再对转换得来的数值作相应的数值运算,就很难根据最后的数值推算出字符值,这就是所谓不可逆函数。

笔者根据这种思想设计了一种加密算法,其核心是:

$KEYS' = KEYS \text{ 长度除以 } 2 \text{ 取整} + KEYS \text{ 的每位字符的 ASCII 码值。}$

程序采用 FOXBASE 实现,由两部分组成程序 1 LOCK 是加密/解密算法,放在钥匙盘上;程序 2 INIT 是系统查对程序,放在系统当前子目录下。两程序通过全局变量 T 和形参 PASSW 传递数据。PASSW 传递的是输入的字符型值 KEYS, T 是 KEYS 经算法 LOCK 运算得来的 KEYS'。

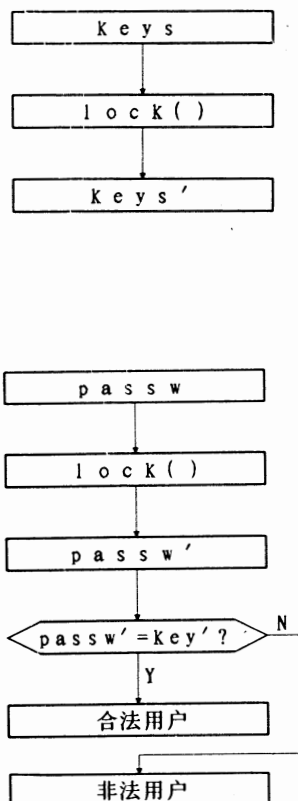
系统查对口令时,要求将钥匙盘插入 A 驱动器,给用户三次输入口令的机会,如果三次口令输入都不正确,则计算机死锁、报警,并在屏幕上显示非法用户。

虽然本程序设计时可公开加密算法,但如果根据加密算法对任意字符串 A 求出对应的值 A',同时改动 INIT 中的 KEYS 为 A',那么输入口令时输入字符串 A,也会被认为是合法操作。因而要妥善保存加密程序 INIT,可以把程序变得很乱,使人无法看出具体算法,或者根本不公布加密算法 LOCK,相信这一点大家都很容易做到。

本程序仅对最普通的单关键值比较进行了说明,有的系统可能要求有多个用户同时使用,其工作权限各不相同,权限表是多维相量时,也可采用这种方法来存储系统密码。

具体实现的不可逆函数可以任意构造,只要它满足要求即可。由于 FOXBASE 提供的 ASCII 码函数可求出汉字的内码值,因而用户口令还可使用汉字,甚至半个汉字,这样就能使系统的安全性进一步提高。

附程序流程图及程序清单



程序1 LOCK

```

PARAMETER PASSW    && PASSW用户输入口令
N=LEN(PASSW)
I=1
T=INT(N/2) && T是加密算法得到的值KEYS'
DO WHILE I<=N
    T=T+ASC(SUBS(PASSW,I,1))
    I=I+1
ENDDO
RETURN
    
```

程序2 INIT

```

PUBLIC T
SET SCORE OFF
SET TALK OFF
SET ESCAP OFF
ON ERROR RETURN
KeyS=321 && 通过加密算法得到的字符串'USER'的值
NO=3 && 用户可输入口令次数
DO WHILE Cond
    IF NO<>0
        @ 15,30 SAY "用户口令:"
        SET CONSOLE OFF
        ACCEPT TO PASSWORD
        DO A:LOCK WITH PASSWORD
        SET CONSOLE ON
        IF T=KeyS
            Cond=.f.
        ELSE
            NO=NO-1
            @ ROW(),40 SAY "口令出错"
            MM=1
            @ ROW(),40 say ""
            DO WHILE MM<10 && 口令错一次报警次数
                ?? CHR(7)
                ?? ""
                MM=MM+1
            ENDDO
        ENDIF
    ELSE
        @ 15,26 SAY "...非法用户 系统死锁..."
        *显示非法用户
        DO WHILE Cond
            @ 15,37 SAY "" && 死锁并报警
            ?? CHR(7)
        ENDDO
    ENDIF
ENDDO
RETURN
    
```

参考文献

【1】《汉字 FOXBASE plus 教程》宗振铎主编,西安交大出版社

用键盘中断解决“死机”问题

吉林电气化专科学校 张 研

对于一个程序员来说,很难使自己编制的程序不经修改就能通过。可见,编制程序调试是一个不可缺少的重要环节。

在调试程序的时候,往往会遇到“死机”现象。“死机”的原因主要是由于程序陷入了死循环,这时就不得不重新启动电脑。这样不仅浪费了时间,而且影响到机器的使用寿命。

能不能在死机的时候用键盘中断死循环呢?笔者试着用汇编语言编写了一段程序(见程序清单)。程序通过修改该键盘中断程序,达到了这一目的。由于键盘中断是 IRQ1,它有很高的优先权,所以可以解决大部分死机问题。

原理说明:大家知道当你每次按下键盘时无论有没有程序在运行都引起一次 INT9H 中断,中断结束返回中断点。当 CPU 接到一个中断请求时,首先将当前有关信息保存在堆栈中(包括状态标志 CS 和 IP),然后将中断入口地址放入 CS, IP 寄存器,这样就可以转入相应的例行程序。返回时,恢复 CS、IP 和状态标志,使能够继续执行刚刚被中断的程序。保存信息的过程如下:

- (1) $(sp) = (sp) - 2$
 $((sp) + 1: (sp)) = flags$
- (2) $(sp) = (sp) - 2$
 $((sp) + 1: (sp)) = cs$
- (3) $(sp) = (sp) - 2$
 $((sp) + 1: (sp)) = ip$

在返回时,过程恰好相反。只要我们修改堆栈中 CS 和 IP 的值,就能在返回时不再回到原来的中断点,而执行用户自己的程序。

本程序就是根据这一原理编写的,程序清单附在文章后面

程序中为了避免程序的重装入,修改了 21H 中断。经过实践,发现该程序不但可以在死机的时候正确返回 DOS,而且还能够中断其它正运行的程序。例如: PE 和 PCTOOLS 等软件。

使用方法:在调试程序前执行本程序,使其驻留内存。当遇到“死机”的时候按 CTRL-B 即可返回 DOS。

附程序清单:

```
code segment
assume cs:code,ds:code
org 100h
start:
jmp begin
int9off dw? ;原键盘中断偏移量
int9seg dw? ;原键盘中断段址
hkl equ 04 ;"CTRL"键
hk2 equ 30h ;"B"键
int2loff dw? ;原21h中断偏移量
int21seg dw? ;原21h中断段址
comp:
push ax ;AX入栈
in al,60h ;读键盘
cmp al,hk2 ;是"B"键吗?
jnz sint9 ;不是转原键盘中断
mov ah,02 ;2号功能
int 16h ;调用键盘I/O中断
and al,hkl
jz sint9 ;"CTRL"键未按下转原键盘中断
int9:
pop ax ;寄存器AX出栈
pop ax ;寄存器AX出栈
push cs ;寄存器CS入栈
mov ax,offset exit ;寄存器AX指向EXIT
push ax ;AX入栈
sint9:
pop ax
jmp cs:dword ptr int9off ;进入原键盘中断程序
exit:
mov ah,4ch ;返回DOS
int 21h ;新的21H中断入口
int21:
cmp ah,0ffh ;调用FF号功能吗?
jnz sint21 ;不是转原21H中断
mov al,0aah ;寄存器AL赋0AAH
iret ;中断返回
sint21:
jmp cs:dword ptr int2loff ;执行原21H中断
begin:
mov ah,0ffh ;调用0FFH功能
int 21h
cmp al,0aah ;安装过否?
jnz install ;未安装过进行安装
push cs ;CS-->DS
pop ds
mov ah,09
mov dx,offset mess1 ;显示已安装信息
int 21h
mov ah,4ch ;返回DOS
;安装程序
install:
mov ax,3609
int 21h ;取键盘中断向量
mov int9off,bx
mov int9seg,es ;存键盘中断向量
mov ax,2609h
mov dx,offset comp ;修改键盘中断向量使指向COMP
push cs
pop ds
int 21h
mov ax,3621h
int 21h ;取21H中断向量
mov int2loff,bx
mov int21seg,es ;存21H中断向量
mov ah,25h
mov dx,offset int21 ;修改21H中断向量使指向INT21
int 21h
mov ah,09 ;调用09号功能
mov dx,offset mess2
int 21h ;显示"OK!"
mov dx,offset begin
int 27h ;驻留并返回DOS
mess1 db "Break-key has been installed! ",0dh,0ah,24h
mess2 db "OK! ",0dh,0ah,24h
ends
end star
```

电子工业出版社广州科技公司

邮购书目

书 名	定 价	邮挂费	出 版 单 位
微计算机的下一场革命——多媒体技术	12.00 元	1.80 元	电子出版社
MS-DOS 结构分析教材之一: BIO 结构分析教程	21.50 元	3.20 元	电子出版社
高频开关稳压电源	9.80 元	1.50 元	广东科技出版社
开关电源集成电路手册	18.00 元	2.70 元	中国电子器材深圳公司 (英文资料)
MICROSOFT C6.0 大全	34.00 元	5.10 元	中国科技大学出版社
亚洲“四小龙”经济教育	13.80 元	2.00 元	电子出版社
NOVELL 网络安装·使用·开发指南(V2.15~V3.11)	15.50 元	2.30 元	电子出版社
NOVELL 网络及其互联技术	13.00 元	2.00 元	电子出版社
NOVELL 网络故障诊断与维修技术	6.50 元	1.30 元	电子出版社
NOVELL 实用通信编程技术	26.00 元	4.00 元	新地文化事业出版
Netware 网络操作指南	22.00 元	3.00 元	新地文化事业出版
Netware 网络应用指南	22.00 元	3.00 元	新地文化事业版
MODEM 调制解调器技术与应用	14.00 元	2.00 元	时代科技信息出版

本期新书介绍

• NOVELL 网络安装·使用·开发指南(V2.15~V3.11)

汤岳清 康奇林 编 李智 审校

开本:16开 276页 定价:15.50元 邮挂费:2.30元

该书作者为从事网络软件开发人员,他们在多年实践经验的基础上,参考大量 NOVELL 最新资料编写而成。为网络购买者、安装者、管理员,使用者和应用程序开发者提供最新、最实用的信息资料。全书共分为五大部分。

第一部分是 Novell Netware 产品综述:分别介绍 Novell 集成计算体系结构以及各种产品;Novell 通信产品;TCP/IP 产品和 Netware 开发工具。

第二部分是 Novell Netware 安装指南:分别介绍 Netware SFT2.15, Netware386 V3.11 的规划,安装方法,网络工作站和打印服务器的安装以及网络设置方法,对 Netware Lite 给出了较详细的介绍。

第三部分是 Novell Netware 用户使用指南:对 Netware386 命令都以实例详加说明,对 Netware 的网络打印作了详细介绍。第四部分是 Novell Netware 网络管理员指南:介绍了 Netware 文件服务器控制台命令及用法等。

第五部分是 Novell Netware 网络应用系统开发指南:介绍 Netware MHs, Netware RPC 以及 Netware 系统接口。C Interface for DOS, C Netware compiler/386 和 Netware SQL/XQL 的具体用法,以及如何在 Novell 网上实现汉字信息处理。

• MODEM 调制解调器技术与应用

汤岳清 蒋月勤 郭启纯 编

开本:16 98页 定价:14.00元 邮费:2.00元

介绍 MODEM 分类及标准;如何选购 MODEM;MODEM 安装及测试;MODEM 在电脑通信中的应用;自己编程控制 MODEM 的方法等。

• MCS-51、8098 单片单板机软件硬件设计

罗维国 吴微 马国敏 编著 武汉大学出版社

16开 192页 定价:9.80元 邮挂费:1.50元

本书以 CYSCB-2 单片单板机为实例,详细介绍其软、硬件设计和具体实现。资料全部公开,便于读者相互借鉴、资料共享。

全书共分七章,内容包括:单片机应用系统与开发系统概述;CYSCB-2 单片单板机硬件分析和监控程序分析,包括与高位机的通信界面。简要介绍了全屏幕编辑和宏汇编软件,最后给出了典型应用程序。

结合本书可提供 CYSCB-2 MCS-51、8098 单片单板机
(售价:880.00元,邮挂费:20.00元)

• 2.13 系列汉字系统用户手册

吴晓军 编著

开本:16 定价:16元 邮挂费:2.4元 晓军电脑工程部出版

本书以 2.13 系列汉字系统最高版本 2.13H 为基础,详细地介绍了 2.13 H 汉字系统的安装启动过程、键盘管理模块、汉字显示模块、汉字打印机模块、实用程序等几个主要方面。最后,还介绍了十种可摘挂到 2.13H 汉字系统上的汉字输出方法。

邮购地址:广州石牌华南师大北区一号 203

邮编:510631

联系人:王丽端

汇款请写明收件人详细地址,姓名,邮编,以便能准确及时地将书籍寄上,需要开发票的请注明;零售请到广州五山路华师大科技服务楼 215。谢谢合作。