中文页面描述语言解释器 CPDL Level 2的设计与实现*

徐福培　金亚东　周　栋　吴　钊

（南京大学计算机科学与技术系，南京 210093）

摘要　CPDL Level 2是一个面向彩色印刷的中文页面描述语言解释器，它与PostScriptLevel 2兼容. 本文介绍了CPDL Level 2的虚拟机模型、层次结构和执行流程，并讨论了它的运行环境和效率.

关键词　页面描述语言，解释器，虚拟机，层次结构.

页面描述语言已广泛应用于电子出版、CAD和办公自动化等领域，因此开展对中文页面描述语言及其解释器的研究，大力推广中文页面描述语言的应用，具有十分重要的意义. 为此，继1990年推出了中文页面描述语言解释器CPDL以后，最近我们又研制成功了性能更完善、面向彩色印刷的CPDL Level 2（简称CPDL 2），目前已成功地安装在某电子出版系统中作为光栅图文处理器，这样不仅可以得到高质量的彩色图文输出，而且也为我国彩色印刷事业步入国际标准创造了条件.

页面描述语言的本质就是把复杂的页面生成过程分解成两个相对独立的步骤来完成. 首先，用户程序利用页面描述语言提供的文字、图形和图象功能将自己要求的页面描述出来，这种描述是与设备无关的；然后，把复杂的、与设备有关的文字、图形和图象的输出还原过程（一系列光栅操作）移交给页面描述语言解释器来完成. 这两步可以在不同时间、不同地点完成，因此解释器的还原质量和效率将是页面描述语言生成高质量输出的关键. 鉴于页面描述语言解释器的特殊地位，结合我们多年研究的体会，提出几条衡量CPDL解释器的标准：

·兼容性好：CPDL解释器必须与PostScript[1]有很好的兼容性，以保证中西文描述和处理与国际标准接轨；

·速度快：这一条至关重要，过慢的页面生成速度将会失去页面描述语言的本来意义，这也是研制开发CPDL解释器的难点之一；

·质量好：这主要是指解释器生成的页面输出其灰度和彩色效果好，彩色无龟纹；中西

* 本文1994－02－04收到，1994－05－27定稿

作者徐福培，1939年生，副教授，主要研究领域为计算机图形学与页面描述语言. 金亚东，1970年生，硕士，主要研究领域为计算机图形学及页面描述语言. 周栋，1970年生，硕士，主要研究领域为计算机图形学及应用. 吴钊，1972年生，硕士，主要研究领域为计算机图形学及应用.

本文通讯联系人：徐福培，南京 210093，南京大学计算机科学与技术系

文小字输出时无断笔和模糊现象;

· 可扩充性、移植性强:CPDL解释器的实现是一项非常复杂、费时的工作,如何使解释器能方便地支持CPDL本身版本的升级,如何使它能方便地从一个光栅设备移植到另一个光栅设备中去,是每个解释器开发者应考虑的问题.

几年来,为了力求达到这些要求,在研制CPDL解释器的过程中,先后对一些关键技术进行了探讨研究,这包括解释器(软解释器和硬解释器)的实现技术、存贮器的管理技术、彩色空间和彩色加网技术、曲线轮廓汉字的自动生成和高速还原技术、汉字的hinting(提示)技术以及中西文字库的组织与管理技术等,且取得了一定的进展,为开发研制高质量的中西文页面描述语言解释器创造了条件.这里仅就解释器的实现技术进行介绍和讨论.

1 CPDL的虚拟机模型

CPDL整个处理过程可以用虚拟机模型来表示(如图1所示).虚拟机顺序解释一个个CPDL语法标记,以状态方式积累信息,并完成成像操作,其结果累积在输出页面上.

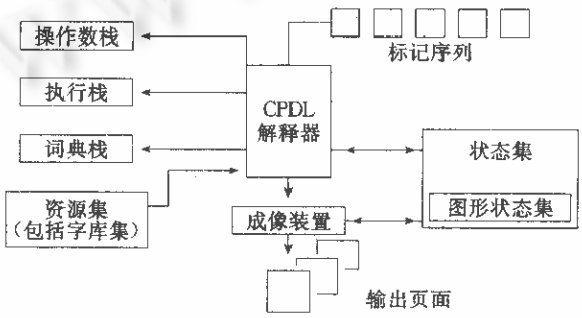

图1 CPDL虚拟机模型

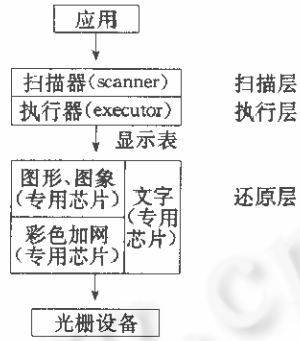

图2 CPDL解释器的层次结构

虚拟机模型由以下几部分组成:

· 标记序列:标记(token)是CPDL语言的语法单位.它被送入CPDL解释器,然后由解释器的扫描程序把一个或多个标记组合成CPDL对象,最后解释器执行这些对象.

· 解释器:它顺序解释标记序列中的每一个标记.如果是一个字面标记,则解释器就把字面标记表示的对象压入操作数栈;如果是一个可执行标记,则解释器就去解释执行与该标记所表示的名字相联系的对象;如果是操作符,则解释器就执行与该操作符相对应的动作.

· 操作数栈和执行栈:操作数栈用于存放操作数和操作结果,执行栈用于存放如过程和文件等可执行对象.

· 词典集:它存放在词典栈中,在CPDL Level 2的实现中,词典栈中常驻内部词典,除了系统词典(systemdict)和用户词典(userdict)外,还有globaldict词典.

· 资源集:资源集是一组命名对象的集合(如中西文各类字库),平时它驻留在磁盘等外设上,需要时才由解释器根据定位策略调入虚拟存贮器(VM).

· 状态变量:状态变量是一个对象,它们可以充当许多操作符的隐含操作数.状态变量中包含一个图形状态变量,存放在隐含的图形状态栈中,它是执行许多图形、图象操作的隐

含控制参数.

· 成像装置:是相应的光栅输出设备,解释器解释执行 CPDL 程序,通过成像装置还原生成图形、图象和文字,它们逐步在输出页面上累积,最后形成所需的输出页面.

2 解释器的层次结构

2.1 层次结构

CPDL 一方面是一个具有较强图形功能的通用程序设计语言,另一方面它又是一个揉合了一般程序设计语言特征的页面描述语言,这两方面是相辅组成的. 因此,CPDL 解释器要完成这两方面的语言解释功能,其结构复杂. 为了便于实现,同时考虑到可扩充性、可移植性,特别是为了便于软解释器的硬件化,并从根本上提高还原速度,这里给出了一个 CPDL 解释器的通用层次模型(如图 2 所示),它由扫描层、执行层和还原层组成.

①扫描层(Scanner)

该层由扫描器(即扫描程序)组成,其主要功能是把由应用程序生成的 SPDL 程序转换成解释器的内部格式,以便提供给执行层解释执行.

CPDL 解释器从外部文件或通信通道获取一字符流作为 CPDL 扫描器的输入,扫描器首先根据分界符从字符流中识别出一个个词法单元(tokens),这相当于一般语言中编译程序的词法分析过程;然后遵循 CPDL 语法规则将一个或多个标记 tokens 装配成一个 CPDL 对象. 标记或对象可以是普通的 ASCII 码正文格式,而在 Level 2 中也可以用"二进制标记编码"和"二进制对象序列"两种格式来表示. 扫描器将三种不同格式表示的对象转换成统一的解释器内部的对象表示格式,并赋于对象相应的属性. 最后扫描层把解释器的内部对象送执行层执行. 这个过程称为 CPDL 的语法分析过程.

②执行层(Excutor)

执行器是始终执行 Execstack 栈顶的对象,一直到栈空为止. 执行器判断对象的可执行属性:若为 False,则将其压入操作数栈;若为 True,则根据对象的不同类型执行不同的动作. 因此可以认为执行器就是 CPDL 语言的语义分析部分.

对于大量的通用程序设计语言的操作符而言,执行层将最终完成相应任务. 而对于文字、图形和图象操作符而言,其动作要复杂得多,而且大量的与具体光栅设备有关的操作还得由还原层来完成.

对于图形,CPDL 执行层将根据程序提供的一系列路径构造操作符,再根据当前变换矩阵 CTM 和其它图形参数(如线型、线端、线宽、线交等),构划出图形轮廓,并由用户坐标空间转换成设备坐标空间,最后生成用实际设备坐标表示的路径描述.

对于文字,CPDL 支持轮廓字库,每个 CPDL 字符(无论是西文或汉字)均对应一段 CPDL 程序. 执行层将从外部资源中取出字库描述,执行相应过程就在当前路径中加入该字符(或汉字)的轮廓表示.

这两种方法构造的路径对于 CPDL 解释器而言是等价的,从而使得汉字在 CPDL 解释器中实际上也作为图形来对等,以便于文字的放大、旋转或进行一些特殊效果的处理.

取样图象是实现世界的反映,其处理方式略有不同,执行器将根据象素阵列,然后送还原层,在具体设备上生成彩色或灰度图象.

无论是文字、图形或图象,执行层还必须把从CPDL程序中读取的彩色空间和彩色值转换成具体光栅设备的彩色空间和相应的彩色值,以便送还原层进行彩色图形的扫描转换和彩色加网处理[2,3].

③还原层

页面描述语言的最终目标就是根据输出页面的大小和设备的分辨率,得到所需页面的象素阵列,并把它们送给光栅输出设备,生成最终的实际页面.因此,还原层的主要功能就是根据执行层送来的路径描述和彩色信息进行扫描和彩色加网的过程,从而从具体物理设备上得到所需的页面输出.

还原层的主要功能涉及到扫描转换规则和彩色加网算法的设计实现,我们之所以强调要将执行层与还原层分开,是因为还原层主要实现与具体设备有关的操作,因此不同性质的光栅设备(如显示器、打印机或照排机),甚至同一类光栅设备的不同分辨率,扫描规则和彩色加网细节都应有不同的要求,以求得到设备支持的最完美的输出效果.

2.2 解释效率的分析

通过以上对CPDL解释器层次模型的介绍可以分析它的执行效率并提出相应的解决办法.第1、2层是CPDL语言的语法、语义分析过程,无论是软解释器或硬解释器,它可以由通用的CPU来完成,但是通过进一步优化内存管理(特别是VM的组织与管理)、简化语义翻译流程、合理的资源配置均能在一定程度上提高执行效率.为此我们多年来针对不同的软、硬件系统环境,提出了VM的管理与调度算法、save—restore的时间戳算法、名字的维护算法、分类词典的组织与管理和资源的调度与定位策略等算法,收到了较好的效果.

第3层是扫描转换和彩色加网过程,其操作与具体光栅设备密切相关,它是解释器效率的瓶颈所在.理论上所有图形、图象和文字操作最后都归结于一个任意多边形的彩色加网过程,通过优化任意多边形的裁剪填充算法和彩色加网算法,就可以从总体上提高执行速度.但是在很多情况下还是难以满足要求,这可以通过专用芯片来实现.在执行层和还原层(专用芯片)之间的接口,可以通过显示表(Display List)来实现.显示表的格式可以根据执行层送给还原层的路径描述和彩色信息以及专用芯片的要求来具体组织.

对于彩色图形和图象的还原处理[4],可以采用Adobe公司专为PostScript硬解释器研制开发的专用芯片"PixelBurst Coprocessor",也可以使用通用的图形芯片(如TMS 34010/34020),甚至采用通用的386/486芯片,经实践证明均能收到良好的效果.而彩色加网也可通过开发专用芯片来实现.

对于文字,由于一个页面描述中往往文字占了大部分,所以其还原速度尤为重要.它既可以使用根据自己的字库组织结构和还原算法开发的专用芯片,也可以采用一些现有的芯片.如Adobe Type1 Coprocessor芯片或Desting公司的RIDA芯片(Universal Font Processor),它们的字符还原速率均可达到2000字符/秒以上,同样适用于汉字,只要向该芯片送入汉字的轮廓描述以及控制汉字大小、方向的变换矩阵,再加上适当的控制信息,就可以得到高质量的汉字输出,且还原速度可达600—1500汉字/秒,这已在硬件解释器上得以实现,取得了良好的效果.

3 软件开发环境

CPDL 解释器功能庞杂,代码量和数据量很大,因此要设计一个优秀解释器的首要任务是选择一个好的软件运行环境,这对于解释器的执行效率和质量至关重要.为此几年来为了开发基于 PC 机的软件解释器,我们先后在 DOS、Windows 和 DOS Extender 环境下开发了多个完整的 CPDL Level 1 和 Level 2 解释器,它们各有优、缺点.

3.1 DOS 环境

DOS 环境具有广泛的可适用性,只要遵循一定规范,在其中开发的程序可以在 286 以上的任何机型上运行,因此早期开发的 CPDL Level 1 和其扩充版可在此环境下运行.但是由于 DOS 本身的缺陷,使得在此系统下运行的解释器,其效率和性能上受到较大的限制:

• VM 是存放 CPDL 复合对象值的地方,它至少必须占用存贮空间 300KB 以上,且 VM 的组织与管理机制的优劣将极大地影响解释器的效率与性能.因此,如果 DOS 环境下且仅有 1MB 内存空间,则 VM 机制就难以实现.如果配有扩展内存,则 VM 机制可通过 XMS 实现.

但由于局部 VM 和全局 VM 的使用频度高,因此常规内存与扩展内存之间的调度频繁;且由于保护方式与实方式的 CPU 模式切换也过于频繁,这些均从总体上影响了解释器的执行速度.

• 为了在物理设备上得到所需输出,必须首先在输出缓冲中形成页面的点阵信息.由于该信息量大,例如,以普通的激光印字机为例,A4 纸 300dpi 分辨率的点阵数据为 1.07MB(黑白输出);如果为了得到彩色输出,则 C、M、Y、K(青、品红、黄、黑)4 个页面的缓存共需 4×1.07 兆字节.在没有扩展内存的情况下,输出缓存只能开辟在硬盘上,在内存辟出一个适当区域(CPDL 1 实现中为 16KB)作为子页,解释器生成的部分页面数据先放在子页缓冲内,然后再写入硬盘上页面的对应区域内.这种逐步生成子页再写入磁盘相应页面缓冲的方法,频繁地读/写硬盘势必大大地影响页面生成速度.当然,如果有 4MB 或 8MB 扩展存贮器,那页面缓冲可以开辟在其间,页面生成速度将大大加快.CPDL 解释器页面缓存的设置可以根据 PC 机 DOS 环境下是否配置有扩展内存以及它的大小自动选择:在无扩展内存情况下,缓存设置在硬盘上;在配有扩展内存的情况下,输出缓存自动设置在扩展内存中.

3.2 Windows 环境

在 Windows 3.0 环境下开发的 CPDL 解释器 WinCPDL 是解决内存瓶颈的一个尝试,其优点是为用户提供了一个良好的用户界面,但是解释器的实际速度没有根本上改变,原因大致有以下几条:

• 采用了 Windows SDK 3.0,该版本实际上是 16 位的,虽然能支持高达 16MB 内存要求,但是每个数据段仍是 64KB,所以不能解决整块页面缓存的要求.

• Windows 是一个多任务的操作系统,支持以句柄方式访问内存,由于 64KB 段长的限制,内存块的锁定与释放十分频繁,这也在一定程度上影响了速度.

• Windows 中所有的图形操作只能通过 GDI 接口完成,但是由于 GDI 3.0 提供的页面坐标精度(16 位整型)远满足不了 CPDL 的要求,大部分 GDI 函数不能使用,CPDL 必须自行处理后再交 GDI 处理,因此速度反而减慢了.

当然,如果采用Microsoft公司新近公布的Win32 SDK版本,其GDI已支持浮点坐标,Kernel也支持32位直接寻址,因此Win32接口的版本(Windows 4.0或Windows NT)将是开发CPDL解释器的最佳宿主环境之一.

3.3 DOS Extender环境

32位DOS扩展器是目前较合适的CPDL解释器开发环境,pharlap的扩展DOS+Metaware High C为应用程序提供了一个类似于UNIX的编程环境而无需担心内存问题,它的优势在于:

- 充分发掘32位CPU的功能,提供了广阔的地址空间,单个代码段或数据段可达4GB;
- 使用32位指令,不需要以前繁琐的段操作;
- 支持虚拟存贮器,利用CPU的paging功能实现内存与外设的一体化;
- 高度优化的编译技术,High C使用了高级的局部优化技术,代码质量高;
- 在pharlap的扩展DOS高版本中支持DPMI接口,从而与Windows以及多种扩充内存访问模式充分兼容,适应性更广泛.

当然,DOS Extender与其它环境相比要求机器性能高(必须386以上),这是其唯一的缺点.综上所述,DOS环境下生成的CPDL解释器受内存束缚的影响大,编程困难,功能扩充与改进大受影响,所以这不是一种值得推荐的工作环境.Win32和DOS Extender是比较理想的开发环境,它们可以充分利用各自的优点,得到性能好、效率高的CPDL解释器.

4 CPDL流程

CPDL解释器的主要功能是对由应用产生的CPDL程序所提供的命令进行解释执行,并在相应输出设备上生成所需页面.要解释的命令可以文件格式提供,也可以交互式提供.

1.总流程

从解释执行这一点出发,CPDL总流程是一个解释流程,如图3所示.

整个流程实际上分成两部分,初始化和不断取出执行栈顶对象反复执行,直到栈空为止.

初始化部分完成运行环境的设置,包括装入驱动程序,申请VM空间,程序文件的打开,装载系统词典和用户词典和命令行参数的解释,判断是交互方式还是文件方式等.

执行部分,解释器不断扫描读入一个语法单位(tokens),并根据其不同属性分别处理:

- 如果是操作数或其它字面名,则压入操作数栈;
- 如果是可执行名,则在解释器的词典上下文中查找与该名字相对应的值,执行相应的内部功能;
- 如果是除了可执行名以外的其它可执行对象,如文件、可执行数组(即过程)、可执行串或操作符,则它首先把该对象压入操作数栈,然后在执行栈中压入操作符exec.接着解释器再从执行栈顶取出对象exec,执行exec操作符,从而完成该可执行对象的相应动作.

2.图形执行流程

在页面上还原输出图形(包括图象与文字)这是CPDL主要的、也是最复杂的过程,为此这里将对图形执行流程作进一步的讨论.

在CPDL程序中图形、图象和文字的描述使用的是用户坐标系,而在实际输出时必须

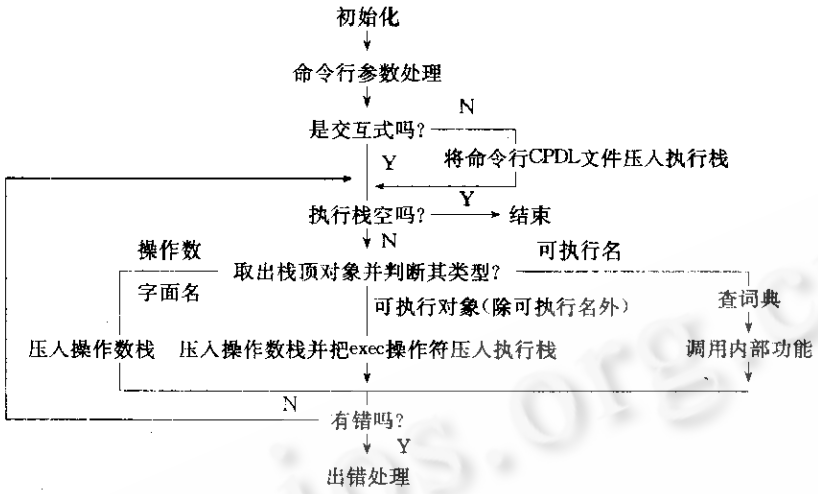

图3 CPDL总流程

转换成具体物理设备的坐标系，这由图形状态中的当前变换矩阵CTM来完成. 当然，CTM中还可以包括对图形对象执行的如平移、旋转、变比等一系列坐标变换. 为了得到多级灰度或彩色的图象，在扫描输出过程中还必须根据灰度值或彩色值进行彩色加网处理(半色调技术). 这里仅给出图形、图象和文字处理的一般流程.

①图形执行流程

图形执行流程可以简单地概括成两个过程：构造路径和扫描转换与彩色加网的过程. 如图4所示.

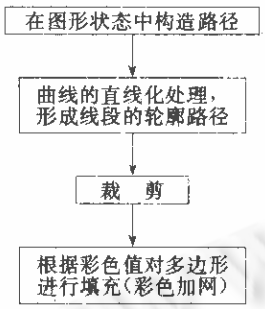

图4 图形执行流程

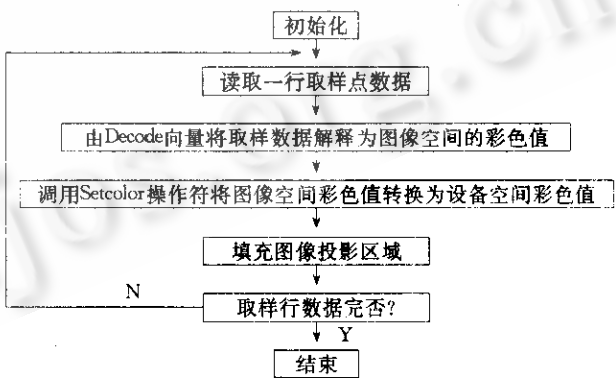

图5 图像处理流程

· 构造路径：执行路径构造操作符(如lineto，arcto，curveto等)，并根据当前变换矩阵CTM的值，在图形状态中形成用设备坐标表示的路径；

· 执行stroke/fill操作符时，首先取出图形状态中的路径描述(如果是曲线则进行直线化)，再根据图形状态中线宽、线型、线端、线交等参数形成直线段的轮廓路径；接着根据当前彩色值形成的彩色网格数据对多边形进行扫描填充，同时完成裁剪功能，最后在页面上生成所需形状的彩色/灰度图形.

②图象处理流程

这比图形流程要复杂一些,特别是 CPDL 2 中彩色图象的处理更有如下一些特点:

· 数据源的多样性,且取样数据量庞大.

· 原图象到目标图象的映射可以是任意变换.

因此,CPDL 图象系统的实现关键在于:取样数据的获得和解释;以及取样点在目标图象的投影区域的填充.

给出图象处理流程如图 5 所示.

由于取样数据量庞大,不可能将整个图象数据同时读入处理,而以一行取样点数据为单位进行读出,并进行取样值到彩色值的转换,接着仍以行为单位填充图象投影区域(矩形或任意平行四边形),最后判断整个图象数据是否已处理完.如果未完,则继续上述流程,直到处理完整个图象数据.

有关字符处理流程另文再作介绍.

5 结束语

CPDL 2 软件、硬件解释器已在新一代的某电子出版系统中得到了应用,并取得了良好的效果,这也为进一步研制开发面向高精度(3000dpi 以上)彩色输出的解释器创造了条件.

参考文献

1 PostScript Language Reference Manual. Adobe System, 1990.

2 Henry McGilton, Mary Campione. PostScript by example. 1992.

3 Roth S F. Real World PostScript, 1988.

1 Foley *et al*. Computer graphics principles and practice. 1990.

THE DESIGN AND IMPLEMENTATION OF CHINESE PAGE DESCRIPTION LANGUAGE INTERPRETER CPDL LEVEL 2

Xu Fupei Jin Yadong Zhou Dong Wu Zhao

(*Department of Computer Science and Technology, Nanjing University, Nanjing* 210093)

Abstract CPDL Level 2 is a Chinese page description language interpreter for color electronic publishing, it is compatible with PostScript Level 2. In this paper, the virtual machine model, hierarchical structure, and flowchart of CPDL 2 are described. The running environment and efficiency of CPDL Level 2 are discussed.

Key words Page description language, interpreter, virtual machine, hierarchical structure.