高等学校计算机教材

中文信息处理实验教程

主审　朱巧明　　主编　朱晓旭

苏州大学出版社
Soochow University Press

中文信息处理实验教程

主审　朱巧明
主编　朱晓旭

苏州大学出版社

图书在版编目(CIP)数据

中文信息处理实验教程／朱晓旭主编. —苏州：苏州大学出版社，2016.5
ISBN 978-7-5672-1735-5

Ⅰ.①中… Ⅱ.①朱… Ⅲ.①汉字信息处理-高等学校-教材 Ⅳ.①TP391.12

中国版本图书馆 CIP 数据核字(2016)第 105815 号

中文信息处理实验教程

朱晓旭 主编

责任编辑 管兆宁

苏州大学出版社出版发行
(地址：苏州市十梓街 1 号 邮编：215006)
虎彩印艺股份有限公司印装
(地址：东莞市虎门镇陈黄村工业区石鼓岗 邮编：523925)

开本 787×1092 1/16 印张 8.75 字数 218 千
2016 年 6 月第 1 版 2016 年 6 月第 1 次印刷
ISBN 978-7-5672-1735-5 定价：22.00 元

苏州大学版图书若有印装错误，本社负责调换
苏州大学出版社营销部 电话：0512-65225020
苏州大学出版社网址 http://www.sudapress.com

·前 言·

中文包括汉字、蒙文、藏文、彝文等多种文字,其中汉字使用最为广泛。中文信息处理是一个很宽泛的概念,本书中的中文信息处理特指用计算机对汉字的音、形、义等信息进行加工和操作的技术与过程。

中文信息处理可以泛指在字符层和内容层两个方面的研究。字符层主要解决中文在计算机中的表示、存储、输入和输出,目前这一领域的研究已经取得了较大进展,涌现出大量的成熟技术和产品。内容层的最终目的是试图理解文本的语义,从而在文本内容上进行高质量的分析和处理。这一层次需求极多,在信息检索、信息过滤、信息抽取、机器翻译、文本情感分析、舆情分析等研究领域也已经取得了一定的进展,但离完美应用尚有很大的差距,该领域的进一步研究具有极大的商业和实用价值。

中文信息处理是计算机科学和语言文字学的交叉学科,它需要综合使用语言文字学、计算机技术、机器学习、人工智能、数学等多种理论和技术。本教程旨在利用计算机对中文信息处理中的一些实用问题进行实验。

本书共分十四个实验,前八个实验主要训练字符层文本信息处理的相关技术,从实验九开始立足内容层进行文本信息处理。

本书是在总结多年教学经验的基础上,汲取了中文信息处理研究领域的新技术和新成果,是在多届学生使用的实验教程讲义基础上修改完成的。本书实验所需要的电子文档和数据可在如下网址下载:http://pan.baidu.com/s/1o82Bm2m,也可以联系 xiaoxzhu@suda.edu.cn 索取。

本书由朱晓旭主编,由朱巧明主审,刁红军参与了实验数据的设计和整理。在此,衷心感谢李培峰、吴娴、赵雷、杨季文等在本书编写过程中提出的宝贵意见和建议。

由于编者水平有限,书中的疏漏之处在所难免,恳请读者批评指正。

编 者

2016 年 2 月

目 录

实验一　汉字在计算机内存储及文件操作

实验目的

1. 熟悉汉字的机内码，掌握以十六进制形式查看文件内容的方法。
2. 理解 GB2312—80 中汉字的分布规律，掌握区位码到汉字机内码的转换规则。
3. 掌握编写程序对文本文件和二进制文件的读写操作。
4. 了解中文信息处理的主要研究领域与研究现状。

实验内容

1. 使用十六进制编辑器查看自己姓名的机内码。
2. 编写汉字区位码查看程序，让用户输入一个汉字从而显示该汉字的区位码。
3. 熟悉文件操作，编写程序将同样的数据分别写入文本文件和二进制文件，比较二者的区别。
4. 了解中英机器翻译系统的研究进展与实用情况。

预备知识

一、ASCII、GB2312—80 与 Unicode

因为计算机采用二进制的方式存储数据，所以包括数值、文字、图片、声音和动画等信息在存储到计算机内时，都必须要转换为二进制。人们使用的十进制数值数据存储到计算机中时，也需要转换为二进制，这个过程相对简单，也较容易理解。但人们使用的文字（尤其是中文）是如何存储到计算机中的呢？

- ASCII

ASCII（美国信息交换标准码）是 American Standard Code for Information Interchange 的

缩写,它是一个用于字符编码的美国标准。ASCII 起始于 20 世纪 50 年代,在 1967 年定稿,标准的 ASCII 采用了 7 位二进制数来表示一个字符,因此至多只能包含 128 个符号。ASCII 中包括了所有的大写和小写英文字母、数字 0 到 9、英文标点符号以及控制字符。

图 1.1 显示了 ASCII 中的符号分布情况,这些符号可以被分为图形字符区和控制字符区。图形字符区合计有 94 个符号,例如,“a”“A”“2”“%”;控制字符区包含了回车、换行、退格等符号。中文的图形字符达到数万个,远远超过 94 个。因此,不能简单地通过对 ASCII 进行修改和定制来解决中文在计算机内的表示问题。

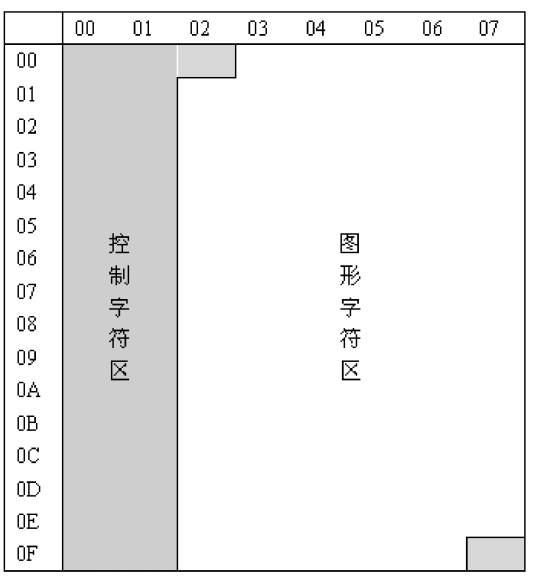

图 1.1　ASCII 中的字符分布情况

- GB2312—80

为了规范、有效地在计算机中处理汉字,我国在 1980 年发布了 GB2312—80 汉字编码国家标准,其全称为“信息交换用汉字编码字符集—基本集”。GB2312—80 采用两个字节来为汉字编码,其编码范围是高位和低位都在 A1H ~ FEH 之间,汉字内码从 B0A1H 开始,结束于 F7FEH,共收录汉字及其他符号 7445 个,其中汉字 6763 个,这些汉字被分为一级汉字和二级汉字两个子集。一级汉字按照拼音排序,合计 3755 个;二级汉字按照偏旁部首排序,合计 3008 个。

GB2312—80 基本满足了当时计算机处理汉字的需要,按照使用频率来看,它所收录的 6763 个汉字已经覆盖了中国大陆大约 99.75% 的汉字使用场合。而且当时的计算机存储空间小、运算速度低,如果使用太大的字符集会导致汉字字形库存储空间巨大,降低信息处理涉及的各种检索速度,不利于使用与推广。

- Unicode

由于很多国家和地区都独立设计了在计算机内表示自己所用文字的编码标准,例如,中国大陆有 GB2312—80,中国香港和台湾地区有 Big-5,它们之间互不兼容,从而导致

不同国家和地区之间文档的交互出现了很多问题。显然,避免上述互不兼容的一个有效措施是全球采用同一编码,每一个字符只对应一个编码,每一个编码只对应一个字符。1991 年,IBM、DEC、Sun、Xerox、Apple、Microsoft、Novell 等公司联合组织了一个协会,创建了 Unicode 项目。该项目的一个主要目标是:无论什么平台、什么程序、什么语言,为每一个字符只提供一个唯一的编码。Unicode 并没有寻找方法来解决兼容当时已经存在的各种编码,而是采取了重新编码的方法。

宏观来看,Unicode 编码系统可分为编码方式和实现方式两个层次。编码方式采用了组(group)、面(plane)、行(row)和位(cell)来给字符编码,组可以有 128 个,而面、行和位都可以有 256 个,所以有 128 × 256 × 256 × 256 = 2147483648 个码位。显然 Unicode 编码空间非常巨大。目前最多被使用的是 0 组 0 面,该面被称为 Basic Multilingual Plane(基本多语言平面),简称 BMP。该平面一共有 256 × 256 = 65536 个码位,基本满足了目前世界上各种常用字符的使用。虽然每个字符的 Unicode 编码是确定的,但是在实际传输过程中,考虑到应用于不同系统的平台以及节约存储空间的目的,对 Unicode 编码的实现方式有所不同。Unicode 的实现方式也被称为 Unicode 转换格式(Unicode Translation Format,简称为 UTF),常见的有 UTF-7、UTF-8、UTF-16 和 UTF-32 等。UTF-8 是目前互联网上使用较多的实现方式,在 Web 网页编码中被广泛使用。

二、GB2312—80 区位码与机内码

GB2312—80 在给汉字编码的时候,设计了一个 94 × 94 的表格,然后在其中对应的位置放上字符,这样通过行列号就可以唯一地确定每一个字符,该行列号也被称为区位号。GB2312—80 中的字符分布情况见图 1.2。

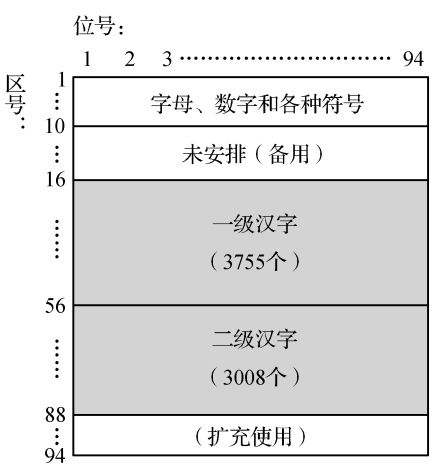

图 1.2　GB2312—80 中的字符分布情况

一个汉字所在的行列号也被称为区位码,区位码通常用十进制数字表示,共 4 位数字,前两位为区号,后两位为位号。如汉字“国”在第 25 区第 90 位,所以它的区位码为

2590。图 1.3 列举了 GB2312—80 的 01 区和 16 区的字符,其中 16 区是第一个汉字符号区。

图 1.3　GB2312—80 的 01 区和 16 区的字符

但是区位码并不适合直接在计算机中存储汉字,如“码”的区位码为 3475,如果直接把 3475 存储到计算机的两个字节中,将来读取时会和“4”与“U”两个字符的 ASCII 码混淆。显然,此时需要一种手段在计算机中能够有效地区分出汉字和西文。计算机分配存储的最基本单位是字节(8 位),但是 ASCII 码通常为 7 位,因此存储时最高位通常为 0。如果保证一个汉字存储在计算机中的每个字节最高位为 1,那么显然就可以很方便地和 ASCII 字符区分出来。因此实现时,把 GB2312—80 中一个汉字区位码的区号和位号先分别转换为十六进制,然后再分别加上 A0H,就得到该汉字的机内码。这样做的优点是很容易区分汉字与西文字符,缺点是实质上缩小了编码空间。根据前面的方法与公式,很容易能计算出“码”的机内码为 C2EBH。

三. 文本文件与二进制文件

文件是操作系统管理数据的有效手段,也是应用程序最常用的持久存储数据的方法之一。从文件的用途我们可以把文件分成:声音文件、动画文件、图片文件等。但就保存的文件格式而言,常把文件分为二进制文件和文本文件。

文本文件中只能存储字符,它在存储介质中存放每个字符对应的 ASCII 码或机内码,二进制文件可以存储各种二进制数据,如声音、视频等。数字 567 在文本文件的存储形式表示为“5”“6”“7”三个符号的 ASCII,转换成二进制为:00110101 00110110 00110111,共 3 个字节。如果计算机中一个整型数据占两个字节,567 在二进制文件中就保存为 00000010 00110111,如果一个整型数据占四个字节,那么 567 在二进制文件中就保存为 00000000 00000000 00000010 00110111。

Windows 所附带的记事本编辑保存的文件属于文本文件。图片、声音、视频等在计算

机内通常都是以二进制文件的形式保存,值得注意的是,Word、WPS 等字处理软件编辑保存的默认的文件存储格式都属于二进制文件。

四、中文信息处理的发展

中文信息处理通常可以分为字符层和内容层两个处理层次。字符层主要解决中文的存储、输入、输出、处理等问题,经过大约几十年的探索与研究,这一层面的主要技术问题已经基本解决,研究也进展到实用阶段。“WPS”“汉王手写输入”“中文 Office”等实用软件已经耳熟能详,给大家的工作、学习带来了诸多便利。内容层的研究目前刚刚起步,也取得了一些进展,并有了一些实用产品,例如,搜索引擎给网络检索带来很多便利,垃圾邮件过滤系统能帮助大家去除骚扰邮件,但这些层面的应用离实用和完美还有很大的距离。

Unicode 的推进与普及有效地解决了中文的跨地区存储问题;语音识别、手写识别、键盘编码输入可以非常好地满足不同用户的输入需求;大多数计算机操作系统从内核上对中文提供了支持;在各种显示和打印设备上,已经可以得到非常精美的中文输出效果;Word、WPS 等字处理软件已成为大家办公、生活的一部分;激光照排系统也已经给整个印刷业带来了革命性的变化。

内容层的中文信息处理应用前景非常广阔,常见的应用需求有:中文信息检索、中文信息分类、中文信息过滤、机器翻译、中文信息抽取、问答系统、中文情感分析等。但是它也有很大的难度,更加具有挑战性。随着互联网的日益发展,研究人员可以很容易地得到海量的真实语料,从而基于这些语料进行研究与拓展,因此目前基于内容层的信息处理很多是基于语料库的统计与挖掘,同时借助机器学习等手段进行处理。这实际上是一种“从大量已知的数据中抽取特征,然后根据这些特征对未知的数据进行判断的方法”。

机器翻译在处理一些真实句子时可能会得到文不对题的翻译结果,但我们不能因为一些极端的例子而否定内容层的研究。目前机器翻译在一些受限领域(如产品说明书翻译)取得了较好的效果,在一些场合也可以起到对人辅助翻译的作用。基于 n-gram 模型的汉字输入法在保持易用的前提下提高了汉字输入速度,中文科技文献检索已经具有较好的效果,中文搜索引擎也给信息爆炸的网络提供了查找数据的工具。但是也应该看到,这些领域可以提高与改善的空间还非常大。

实验原理

1. 在记事本中输入字符后,可以选择以不同的编码保存。但文本文件最终还是以二进制的形式保存在计算机中,可以用二进制编辑器查看保存的文件,从而加深对字符编码的认识。

2. 通过 GB2312—80 设计的区位码,可以唯一地确定每一个被收录的汉字,但如果

汉字直接以区位码保存在计算机中却是不可行的,因为汉字的区位码可能会和西文的ASCII 码混淆,所以要以机内码的形式存储汉字。

区位码通常用十进制表示,而汉字的机内码往往用十六进制表示,我们只需要把区位码的区号和位号分别变成十六进制,然后分别加上 A0H,就可以得到该汉字的机内码。

3. 文件的存储形式主要有文本文件和二进制文件。但由于计算机是基于二进制存储数据的,文本文件也可以用二进制的方式打开。

可以将同样的数据以文本形式和二进制形式保存,然后用十六进制编辑器打开文件查看,体会二者的区别。

文件操作的首要前提是打开文件,C 语言中可以通过库函数中的 fopen 函数打开文件,文件使用完成后应该关闭文件,此时需要使用 fclose 库函数。

库函数中的 fscanf 和 fprintf 函数可以方便地读写文本文件,而使用 fread 函数和 fwrite 函数可以便捷地对二进制文件进行读写。

C ++中提供了文件流的机制来对文件进行操作,主要有文件输入流(ifstream)、文件输出流(ofstream)和文件输入输出流(fstream)。它们都可以便捷地读写文本文件和二进制文件,具体细节可以参见相关书籍。

实验环境

一、操作系统

Windows 2000 以上版本,例如,Windows 2000/XP/Vista/7/8/8.1/10 等。

二、开发环境

Visual Studio 6.0 以上版本,例如,Visual Studio 6.0/2003/2005/2008/2010/2012/2013 等。

三、十六进制文件编辑器

例如,UltraEdit、Winhex、Hex Workshop 等。

实验步骤

一、查看自己姓名的内码

1. 打开 Windows 附带的记事本。

2. 在记事本中使用汉字输入法输入自己的学号与姓名。

3. 单击【文件】→【保存】项,选择一个路径,以“name. txt”文件名保存该文件,在“另存为”对话框中“编码”项中选择“ANSI”,然后单击“保存”按钮。如图 1.4 所示。

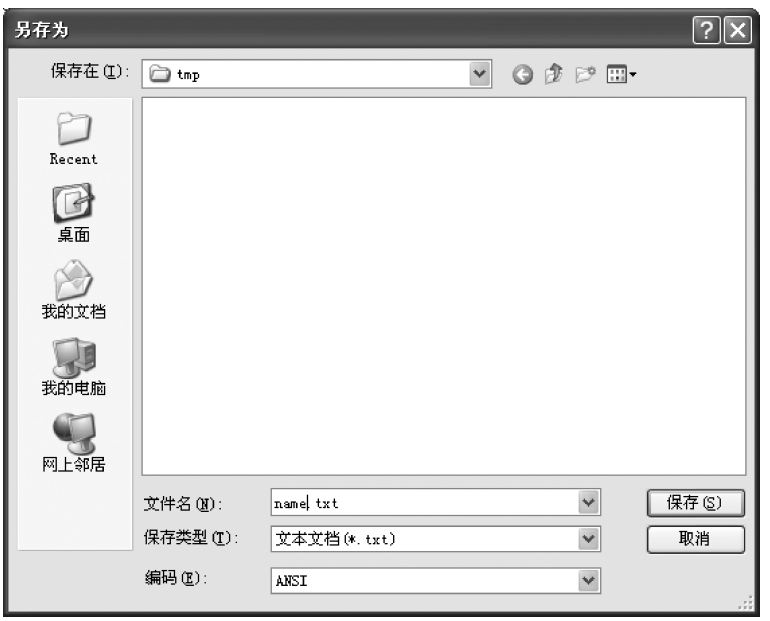

图 1.4　以 ANSI 编码保存文件

4. 关闭记事本程序。

5. 在资源管理器中切换到“name. txt”所在的文件夹,右击“name. txt”文件,单击【属性】项,根据对话框的信息记录下该文件大小。

name. txt 的文件大小(单位:字节)	

6. 用十六进制编辑器(如 UltraEdit 软件)打开“name. txt”文件,记录下该文件的十六进制内容。

name. txt 的内容(十六进制)	

7. 关闭十六进制编辑器。

8. 在记事本中重新打开 name. txt 文件,单击【文件】→【另存为】项,输入文件名“name1. txt”,在“编码”项中选择“Unicode”,然后单击“保存”按钮。

9. 用十六进制编辑器打开“name1. txt”文件,记录下该文件的内容。

name. txt 的内容(十六进制)	

10. 仔细比较两次文件内容的区别,思考为什么同一字符以不同编码保存就不一样?

二、编写汉字区位码查看程序

1. 在 Visual C ++集成环境中新建一个“Win32 Console Application”工程,工程的名称为“ex1_2”。图 1.5 是在 Visual Studio 2010 中建立 Win32 Console Application 项目的界面。

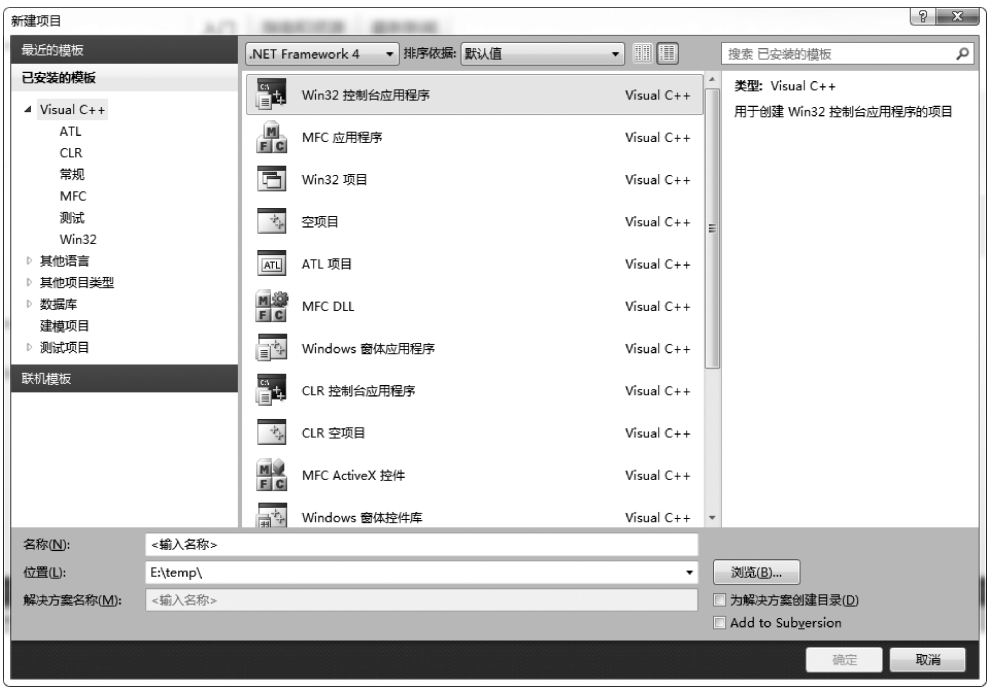

图 1.5 在 Visual Studio 2010 中新建 Win32 Console Application 项目

2. 在源程序编辑窗口编写代码,程序中定义一个具有三个元素的字符数组,然后接受用户输入一个汉字,接着将汉字的两个字节分别减去 A0H,最后以十进制输出,就可以得到该汉字的区位码。将源程序书写到实验报告中。

3. 运行程序,根据运行结果填写下表:

输入	国	码	国家	AB
输出				

三、熟悉文件操作

1. 在 Visual C++集成环境中新建一个“Win32 Console Application”工程,工程的名称为“ex1_3”。

2. 在程序编辑窗口输入程序,以文本方式创建文件“text.txt”和二进制方式创建文件“bin.dat”,产生 10 个 1~100 之间的随机正整数并分别写入这两个文件,其中文本文件中每个整数写一行。将程序填写在实验报告中。

提示: C 和C++语言的库函数 rand()可用于产生 0~RANDMAX 宏之间的一个随机整数。在程序前面添加 #include <stdlib.h> 后就可以使用该函数。函数原型如下:

```
int rand( void );
```

3. 用十六进制编辑器打开两个文件,比较二者的区别,然后根据自己的理解在实验

报告中描述产生区别的原因。

四、了解汉英翻译系统的进展

1. 通过互联网查找一个或多个常用的汉英(英汉)翻译系统,并在实验报告中写出其名称。

2. 有如下英文语句:He saw a duck with a telescope. 。请根据你的理解,写出它可能的翻译。

3. 写出你检索到的不同机器翻译系统对第 2 条中的句子的翻译结果。

4. 自己组织一些句子测试你找到的机器翻译系统,并在实验报告中写出你对目前汉英(英汉)翻译系统的评价。

思考题

1. 编写一个程序,实现 GB2312—80 汉字区位码到机内码的转换。

2. 编写一个程序,实现用户输入 GB2312—80 的区号,输出该区所有的汉字。

3. 编写一个程序,以二进制方式打开一个指定的文本,将文件中的数据顺序读出并写入另外一个文件中,从而实现文件复制的功能。

4. 修改思考题 3 的程序,将读出的数据的每一个字节作一些变化,再写入另外一个文件中,从而实现简单的文件加密。

参考文献

[1] (美)Richard Johnsonbaugh, Martin Kalin 著. ANSI C 应用程序设计[M]. 杨季文,吕强,译. 北京: 清华大学出版社, 2006.1.

[2] GB 2312—1980 信息交换用汉字编码字符集—基本集　http://www.sac.gov.cn/SACSearch/search?channelid=97779&templet=gjcxjg_detail.jsp&searchword=STANDARD_CODE='GB2312-1980'&XZ=Q.

[3] GB 18030—2005 信息技术中文编码字符集　http://www.sac.gov.cn/SACSearch/search?channelid=97779&templet=gjcxjg_detail.jsp&searchword=STANDARD_CODE='GB 18030-2005'&XZ=Q.

[4] GB 13000—2010 信息技术 通用多八位编码字符集(UCS)　http://www.sac.gov.cn/SACSearch/search?channelid=97779&templet=gjcxjg_detail.jsp&searchword=STANDARD_CODE='GB 13000-2010'&XZ=Q.

[5] 刘群. 统计机器翻译综述[J]. 中文信息学报, 2003, 17(4):1-12.

[6] 王继成,萧嵘,孙正兴,等. Web 信息检索研究进展[J]. 计算机研究与发展,2001,02:187-193.

[7] 马亮,陈群秀,蔡莲红. 一种改进的自适应文本信息过滤模型[J]. 计算机研究与发展,2005,01:79-84.

实验二 简-繁(繁-简)汉字机内码的转换

实验目的

1. 深入理解汉字在计算机内的存储形式,熟悉 GBK、Big-5 和 Unicode 中汉字的分布与编码规律。

2. 理解简-繁(繁-简)汉字转换的需求、应用与困难。

3. 掌握简-繁(繁-简)汉字机内码转换的技术。

实验内容

1. 设计一个 ISO10646-1:1993/Unicode 内码和 GBK 内码的转换表结构。

2. 编写程序构建 ISO10646-1:1993/Unicode 内码和 GBK 内码的转换表。

3. 编写应用程序,实现 ISO10646-1:1993/Unicode 内码和 GBK 内码在文件级的转换。

预备知识

一、ISO10646-1:1993/Unicode、GBK 与 Big-5

- GBK

GBK 全称“汉字内码扩展规范”,英文名称为 Chinese Internal Code Specification,由中华人民共和国全国信息技术标准化技术委员会于 1995 年 12 月制定。GB 即“国标”,K 是“扩展”的汉语拼音第一个汉字的第一个字母。

GBK 在编码上与 GB2312—80 兼容,在字汇上与 ISO10646 兼容。实质上,它是一个用于从 GB2312—80 向 ISO10646 过渡的产物。

与 GB2312—80 一样,GBK 也采用了双字节表示,总体编码范围是 8140H ~ FEFEH,

首字节在 81H ~ FEH 之间,尾字节在 40H ~ FEH 之间,剔除了 xx7FH 一条线。GBK 总计有 23940 个码位,共收录了 21886 个汉字和图形符号,其中汉字(包括部首和构件)有 21003 个,图形符号有 883 个。

GBK 的编码结构如表 2.1 所示。汉字的内码分布在 GBK/2(B0A1H ~ F7FEH)、GBK/3(8140H ~ A0FEH)、GBK/4(AA40H ~ FEA0H)中,共有 21003 个。

表 2.1　GBK 的编码结构

类别简称		范围(十六进制)	码位	字符	字符与备注
符号标准区	GBK/1	A1A1 ~ A9FE	846	717	图形符号,GB2312 为主
	GBK/5	A840 ~ A9A0	192	166	图形符号,Big-5 和结构符
	小计	1038	883	图形符号	
汉字标准区	GBK/2	B0A1 ~ F7FE	6768	6763	汉字,GB2312
	GBK/3	8140 ~ A0FE	6080	6080	汉字,GB13000
	GBK/4	AA40 ~ FEA0	8160	8160	汉字,GB13000 等
	小计	21008	21003	汉字	
自定义区	1 区	AAA1 ~ AFFF	564		
	2 区	F8A1 ~ FEFE	658		
	3 区	A140 ~ A7A0	672		限制使用
	小计	1894			
总　计			23940	21886	

- Big-5

Big-5 码是一个通行于台湾、香港地区的繁体汉字编码方案,它被俗称为“大五码”或“五大码”。1984 年,台湾地区 13 家厂商试图合力开发“文字处理”“数据库”“电子表格”“通讯”和“绘图”五大套装软件。虽然开发出来的软件未获成功,但为开发五大套装软件设计的 Big-5 码却成为了业界的主流,后来发展成为了繁体中文使用地区的标准。

Big-5 码共收录汉字 13053 个,它包含 5401 个常用字和 7652 个次常用字两部分。常用字包括台湾地区教育管理部门颁布的《常用汉字标准字体表》中的全部汉字 4808 个,台湾地区中小学教科书常用字 587 个,异体字 6 个;次常用字包括台湾地区教育管理部门颁布的《次常用汉字标准字体表》的全部汉字 6341 个和《罕用汉字标准字体表》中使用频率较高的 1311 个汉字。

Big-5 同样是一个双字节编码,高位字节在 81H ~ FEH 之间,低位字节在 40H ~ 7EH 以及 A1H ~ FEH。

需要注意的是,应该是由于工作中的疏忽,Big-5 中重复收录了两个相同的字。“兀”有 A461H 和 C94AH 两个编码,“豈”也有 DCD1H 和 DDFCH 两个编码。因为在文件和互联网上检索字符往往需要作字符内码的精确匹配,一个字有多个内码会导致查询结果的

查全率很差。

- ISO10646-1:1993/Unicode 中的汉字分布

ISO10646-1:1993/Unicode 的 BMP 平面中从 4E00H 到 9FFFH 是中日韩认同的表意文字区,合计收录了 20902 个中日韩汉字。

二、Unicode 与文本文件

GB2312—80 中每个汉字的机内码由于最高字节一定为 1,因此可以有效地和 ASCII 区分开来。但 Unicode 内码的机内码并没有类似的明显区分方法,因此如果将一段汉字以 Unicode 内码存储到文本文件中,将来在读取的时候很有可能误将其中的数据按照 ANSI Code 或者 ASCII 进行解析,这样同样可能会导致“乱码”。

在 Windows 的文本文件中是通过添加特殊标记文件头来解决该问题的。Windows 操作系统附带的记事本程序在存储 Unicode 文本文件时,会在文件最前面存储了一个 FFFEH 标记。该标记占用了两个字节,接着在后面存储正常文本每个字符的 Unicode 内码。由于 FFFEH 码位在 ANSI Code 和 Unicode 中都没有被使用,因此不存在二义性,这是一种很简单有效的区分 Unicode 与 ANSI Code 文本文件的方法。

三、汉字的简-繁(繁-简)转换的需求

个人、企业、组织和政府部门都有大量的简-繁(繁-简)汉字转换的需求。例如,中国内地的电子邮件使用者,将一个采用 GBK 编码存储汉字的邮件发送给香港地区的用户,那么对方很有可能会看到一个充满了“乱码”的邮件。图 2.1 显示了同一邮件采用不同编码查看而出现的“乱码”。此外,很多跨境的组织机构、公司希望自己网站上发布的最新信息和新闻能够便捷地实现简-繁体同步更新,同样大量软件公司在开发国际化软件的时候也希望自己的软件能够同时支持简繁体中文。

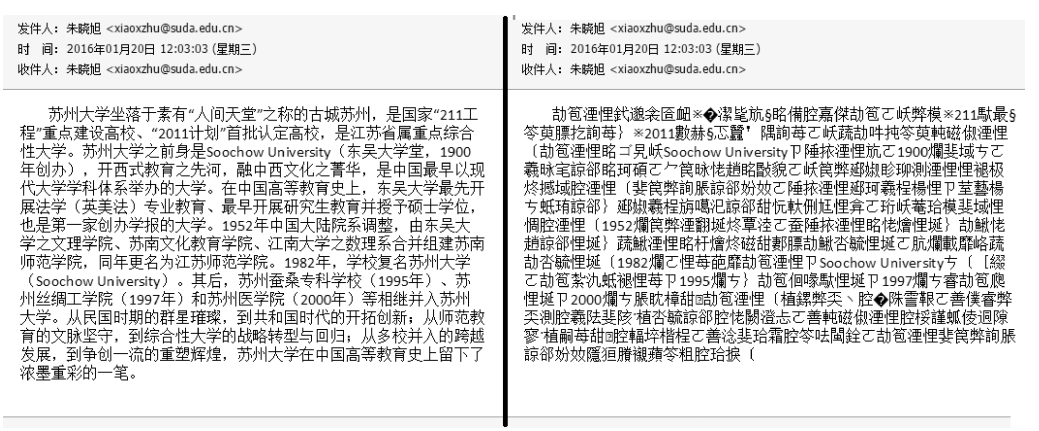

图 2.1 GBK 编码的邮件采用 Big-5 编码查看时出现“乱码”

汉字简-繁转换需求大致可以分成三个层次,第一个层次是内码层,保证在简(繁)体