繁简体汉字按拼音与混合字符按字典序排序

何明昕[1]，炊向军[1]，李家禹[1]，陈宗彝[2]

（1. 暨南大学 计算机科学系，广东 广州 510632；2. 中港通旅运有限公司，广东 深圳 518054）

摘 要：针对编程环境中对繁简汉字按读音排序工具的缺失，根据应用开发需要，通过构建全半角字符的字典序和汉字按拼音排序的<字符 序位码>表，在 Java 环境实现了对汉字按常用读音及全半角字符按字典序的字串比较器类。覆盖 GBK 汉字集，可部署到 JEE Web 服务器供多个并发用户同时使用，也可直接用于独立的 Java 程序，满足了内地及港澳台繁简体中文及中英文混杂信息的相关排序要求。程序在中港车务调度系统中实际应用，获得满意的时空效率及期望的排序结果。该方案采用方法可容易地推广到其他编程环境中。

关键词：汉字排序；拼音；混合字符排序；字典序；全角字符；半角字符

中图法分类号：TP391.1；TP391.12 **文献标识号**：A **文章编号**：1000-7024（2014）06-2009-04

Sorting of Chinese characters by pinyin and combined symbols in lexicographical order

HE Ming-xin[1], CHUI Xiang-jun[1], LI Jia-yu[1], CHEN Zong-yi[2]

(1. Department of Computer Science, Jinan University, Guangzhou 510632, China;
2. ChinaLink Express Limited Company, Shenzhen 518054, China)

Abstract: Due to the absence of sorting tools in programming environments for traditional and simplified Chinese characters by pronunciation, a string comparator that compare Chinesed characters by pinyin with single-byte and double-byte symbols in lexicographical order to satisfy the application requirement. It was implemented in Java environment through a <character order-code> table in which combined symbols in lexicographical order and Chinese characters in pinyin order. It covered GBK Chinese characters set, and could be deployed to JEE Web servers for multiple concurrent users usage at the same time, and could also be used directly in independent Java programs. It satisfied the related requirements of sorting information represented by traditional and simplified Chinese characters with or without single-byte and double-byte symbols. The program had been put into pragmatic usage in the China-HK bus fleet scheduling system and space and time efficiency with expected sorting results were obtained. The methods used in the project could be easily generalized to other programming environments.

Key words: Chinese characters sorting; pinyin; combined symbols sorting; lexicographical order; double-byte character; single-byte character

0 引 言

本文的研发源自于在中港两地使用的企业应用软件开发的实际需要。应用要求针对繁简体混合的中英文信息，按汉字读音顺序对中文信息排序；对其中混杂的英文字符及常用符号，含半角字符与全角字符混合的情形，在要求中文按拼音排序的同时，全/半角混合的信息字串要求按字典序，即 aaAAbbBBccCC 的顺序排列。

在汉字编码及内码实现中，仅 GB2312 中的 3755 个一级汉字是按读音排列的。在 GB2312 及其扩展 GBK 编码系统中，直接用内码对汉字字串排序，多数情形下大多数条目都是按读音排序的，带非一级汉字的条目例外。对要求不严格的一般应用，按 GB 内码排序，通常靠后的以二级汉字居首的条目甚少，基本可达到准读音排序的效果。这

收稿日期：2013-09-02；**修订日期**：2013-11-10

基金项目：国家科技部科研院所技术开发研究专项基金项目（2012EG124225）

作者简介：何明昕（1963-），男，湖北荆州人，博士，副教授，研究方向为软件工程、并行分布式网络计算及企业应用系统；炊向军（1987-），男，河南洛阳人，硕士研究生，研究方向为软件工程；李家禹（1988-），男，天津人，硕士研究生，研究方向为软件开发与测试；陈宗彝（1966-），男，香港人，硕士，研究方向为商业法律、应用会计、企业管理。E-mail：cxjhap@163.com

可能是系统和工具开发商较少关注解决相关问题的主要原因。

由于内地及港澳台使用简体和繁体汉字的习惯差异，在混杂着大量繁体字的信息中，按 GBK 内码排序不能满足实际需要[1,2]。因 GB2312 不包括的常用繁体字在 GBK 内码中分布在一级字库之前以及一二级字库之间，类似以姓名信息为主的条目，以 GBK 码排序，只有中间一块是按读音排序的，靠前的以及靠后的条目不少，很难查找。

目前流行的开发环境普遍缺乏支持繁简汉字按拼音及全半角字符混合字串按字典序排序的工具。在搜索到的少量文献中[3-6]，分别介绍了在 Windows 等特定系统下针对 SQL Server 等特殊环境下的汉字拼音码获取方法及汉字按拼音排序的方法，其适用范围相对狭窄，也难以满足混合字符按字典序排序的要求。

因应用在中港两地使用的车务调度系统的实际需要，在 Java 环境下，针对 UTF-8 编码，覆盖 GBK 字符集中的 2 万多个汉字（包括偏旁），本文实现了繁简体汉字按读音及全半角混合字符串按字典序排序的程序，在应用开发中使用，取得满意效果。

1 字符编码与字典序相关背景

本节先介绍包括汉字在内的字符编码以及西文字符按字典序排序的相关背景知识。

1.1 汉字编码简介

计算机以编码为基础处理字符。ASCII 码以首位为 0 的字节表示英文数字及常见符号，称单字节（半角）字符，高位为 1 的字节可表示扩展的 ASCII 字符，如 DOS 环境中的制表符等。

国标 GB2312-80 按汉语拼音顺序依次按区位排列存放 3755 个一级（最常用）汉字（16-55 区），按部首顺序存放 3008 个二级汉字（次常用）汉字（56-87 区），共 6763 个。GB2312 内码兼容 7 位 ASCII 码，以双字节表示，区/位码分别加 A0，高字节从 B0-F7，低字节从 A1-FE。内码中还包含若干（全角）符号。

GBK 内码遵循国标 GB18030，兼容 GB2312 码。其双字节总体编码范围为 8140-FEFE，首字节 81- FE，尾字节 40-FE，排除 xx7F，共收入 21886 个汉字和图形符号，其中汉字（含部首）21003 个，图形（全角）符号 883 个。

在港澳台地区，广泛使用面向繁体汉字的 BIG5 内码，是与 GBK（或 GB2312）互不兼容的独立编码。

在 Unix 及 Java 环境中，通常使用 Unicode 编码同时处理多国文字字符，有 UCS2，UCS4 等多种编码表示方案以及 UTF-16，UTF-8（变长）等内码实现方案[1,2]。本文工作使用 Java 环境和 UTF-8 内码。

1.2 西文字符的字典序顺序

按字符的内码排序，得到的结果通常是 ABC…abc…的顺序，而按照传统的英文辞典习惯，应该是 aAbBcC…的顺序。不仅如此，当前面的字串字母相同只是大小写差别（称为次级差别，minor difference，而不是字符的本质差别）的时候，要看后续字串的比较结果。例如应是 {adam，Adam，as，As}，而不是 {adam，as，Adam，As}。

Java 中的 java. text. Collator 类[7]提供了处理各国语言的字符按字典序比较的功能，为处理常见的西文字符串比较提供了便捷有效的手段。但 Collator 类仅限于对常见的西文，对汉字按某种规则（拼音或笔划）比较未能支持。对汉字与西文混合的字串，西文按字典序比较也无法使用。

另一方面，汉字字符集中还包括各类全角字符，除常见的英文字母和数字外，还包括一些特殊的符号，如@ & $ ¥以及特殊的拼音注音字符。考虑到全半角字符之后，字符 A 的次级同类字符可能包括@@aaāáǎàAA。

2 排序方案与序位码表设计

Java 提供了丰富高效的数据结构及相应的排序方法，只要设计一个实现 Comparator<T，T>接口的类[7]，类似 Collator 类，就可以使用 Collections 的排序功能。Comparator 接口的核心是其对象比较方法：int compare（T e1，T e2）。本文的核心工作，就是针对带繁体汉字并混合全半角字符的 String 对象，设计实现汉字按读音顺序及字符按字典顺序的字串比较方法 compare（String，String）。

在 Java 内部使用 Unicode 编码处理字符，其基本类型 char 及封装对象 Character 将基本 ASCII 字符（半角字符）与汉字及图形字符（全角字符）统一对待，屏蔽了直接处理字节码的细节和麻烦。因此字串比较依赖于字符逐个比较的结果及其后缀串的比较结果。对于汉字字符，一个字的差异便决定了字串比较结果；对非汉字的全半角字符，对只存在次级差别的前缀，由其后缀字符的本质差异决定；若后缀不存在本质差异，则由其前缀的次级差异决定。

本文的比较算法基于一个完整的包含常用全半角字符和汉字的<字符 序位码>对照表，其中序位码是一个整数。此表由两部分构成：第一部分是按本质差异及次级差异依次排列的半角及全角字符，其序位码由其本质符号序码（前缀码）及最后一位次级序码组成。第二部分是按照常用读音依次排列的汉字及读音序位码。

第一部分全半角字符的序位码由手工编辑而成，目前收集了 60 组共 219 个全半角符号。因序位码采用间隔式编码，可以方便地扩充。

第二部分是汉字序位码表，根据汉字的常用读音依次排列。汉字序位码表是通过以下步骤构造完成的：

（1）获取完整 GBK 汉字的拼音码表，是以拼音码（以 v 代 ü，附加 12345 标识声调）为键值的汉字列表文本文件。

（2）利用 UltraEdit 编辑器将 GBK 编码的拼音码表转换

成 UTF8 内码的拼音码表。

(3) 编制程序将原始的<拼音码 汉字集>文件转换成<汉字 拼音码集>文件，其中的汉字按照（首）拼音码＋汉字内码排序。

(4) 参照文献 [8-10] 对 4389 个重音字做了人工校对，把期望的读音放在拼音码集的首位，例如：将“曾 ceng2 zeng1”修正为“曾 zeng1 ceng2”，因为“曾”在企业应用较多出现在姓名中。

(5) 按照（首）拼音码＋汉字内码排序对码表文件再次排序，以校正重音字修正读音后的位置。

(6) 手工将常用的繁体字调整到与之对应的简体字之后，例如陈陳，杨楊，孙孫，铁鐵等，使繁简混合的相同信息能够在排在邻近的位置。

(7) 编制程序，按（首）拼音码＋汉字条目行号对文件第三次排序，以确保汉字按拼音并校正繁简汉字位置调整中可能出现的意外移位。

(8) 编制程序将码表文件转换为<汉字 CN_OFFSET＋行号>即<汉字 序位码>的格式；其中 CN_OFFSET 人为设置为 8000，目的是为了给全半角字符留下充足的序位码空间。

(9) 将符号序位码表和汉字序位码表合并，便得到完整的<字符 序位码>对照表 cnorder.txt。其中的片段如图 1 所示。

```
y 3960
 y 3961
Y 3962
 Y 3963
￥ 3964
z 3970
 z 3971
Z 3972
 Z 3973
吖 8001
啊 8002
錒 8003
锕 8004
阿 8005
嗄 8006
```

图 1 字符汉字序位码表

3 核心字串比较方法

符号汉字混合字串比较算法以方法的形式封装在实现了接口 Comparator<String，String>的 PinyinCollator 类中。该类定义了静态的类型为 TreeMap<Character，Integer> 的 Map 对象 cnOrder。

核心的 compare 字串比较方法如图 2 所示。

4 实现技术与应用示例

本节介绍在 Web 环境下实现 PinyinCollater 的相关技术，并以实例展示应用接口的使用。

```
int compare (String s1, String s2) {
  int len1 = s1.length();
  int len2 = s2.length();
  int minLen = Math.min(len1, len2);
  if (minLen==0) return len1-len2;

  int minorCompare = 0;
  for (int n=0; n < minLen; n++) {
    int i1 = cnOrder.get(s1.charAt(n));
    int i2 = cnOrder.get(s2.charAt(n));
    if (i1==i2) continue;
    if (i1>CN_OFFSET || i2>CN_OFFSET)
      return i1 - i2;
    if (i1/10 != i2/10) return i1-i2;
    else minorCompare = i1%10 - i2%10;
  }

  if (len1 == len2) return minorCompare;
  if (len1 == minLen) return -1;
  if (len2 == minLen) return 1;
  return 0;
}
```

图 2 字符比较方法 compare 算法梗概

4.1 单实例模式实现序位码对照表

PinyinCollator 是在 Web 环境（本文使用 Tomcat7，也适用于其他 JEE 环境）下供多个并发用户同时使用的，20000 多字的序位码表空间占用的内存不算很大，也不是小数，每次用户调用都去读取码表文件，时间效率也不经济。

本文借鉴单实例设计模式的思想和实现方法，将 TreeMap<Character，Integer> 对象 cnOrder 设计成静态成员，在构造 PinyinCollator 实例对象时，先检查 cnOrdesdr 是否已经构建，这样多用户反复使用也只需要读序位码文件并构建 cnOrder 一次，从提升了时间和空间效率。

根据作者在几台不同性能的微机上的实验测试，初次构建 PinyinCollator 对象（包括了读序位码文件并构建 cnOrder 对象）的时间为 23-60ms，再次调用 PinyinCollator 对象的 compare 方法的时间小于 1ms。

4.2 非页面环境获取 ServeletContext 对象

PinyinCollator 最终要伴随 Web 应用部署到相应的 JEE Web 服务器中，其工作目录会应服务器而变化。而文件读写需要文件存储的路径信息，在 Servlet 容器中的 jsp 文件或 Servlet 环境中可使用 ServletContext 对象的 getRealPath(" /") 获得。在普通的 Java 类中，不能直接访问容量环境对象。

本文设计了一个 ServletContextHolder 静态单实例类，以及一个名为 ServletContextHolder- Listener 的 ServletContextListener，在其 contextInitialized 方法中利用 ServletContext- Event 获得 ServletContext 对象并将其保存到 ServletContextHolder 中，如图 3 所示。

```
public class ServletContextHolderListener
implements ServletContextListener {
  public void contextInitialized (
    ServletContextEvent sce
  ) {
    ServletContext ctx =
         sce.getServletContext();
    ServletContextHolder.
        setServletContext( ctx);
  }
  ...
}
```

图 3 获取 ServeletContext 对象

使用 web.xml 将 ServletContextListener 布置到 Servlet 容器中。在 PinyinCollator 的初始化中调用 ServletContextHolder.getServletContext () 可获得相应环境对象，再调用 getRealPath (" /") 可获得相应目录。

4.3 应用示例

一个对 List<Driver>对象集合中的司机信息按汉字拼音及字符字典序排序的应用如图 4 所示（为节省篇幅，其中的 Driver 简写为 D)。

```
... static PinyinCollator collator;
... List<D> sortByName (List<D> ds) {
  List<D> result = ds;
  Collections.sort( result,
    new Comparator<D> () {
      public int compare (D d1, D d2) {
        if (collator == null)
          collator = new PinyinCollator();
        return collator.compare(
          d1.getName(), d2.getName()
        );
      }
    }
  );
  return result;
}
```

图 4 PinyinCollator 应用示例

5 结束语

通过构建半角及全角字符的字典序和汉字按拼音排序的序位码表，在 Java 环境实现了对汉字按常用读音及全半角字符按字典序的字串比较器类，覆盖了 GBK 汉字集，能满足内地及港澳台繁简体中文及中英文混杂信息的相关排序需求。

本文的程序可直接应用于独立的 Java 程序中，也可部署到 JEE 服务器供多个 Web 并发用户同时使用。本文的成果在中港巴士车务调度系统中得到实际应用，获得满意的时空效率及期望的排序结果。本文使用的方法可容易地推广到其他编程语言环境中。

由于汉字读音的多样性（重音字）在不同的应用中的要求不尽相同，比如本文的应用背景较多地用在姓名，客户及公司名等信息排序中，在构造序位码表时采用常用姓名读音优先的原则，在其他的应用中可能优先考虑其他因素。在不同类型的应用中，应根据实际需要构建具个别细微差异的汉字序位码表。

本文使用的中英文字符排序是在应用服务器中完成的，理论上可应对较大数据量的排序要求。根据作者的经验，在 web 应用架构中，仅适用于中等数据量（比如数万条记录）以内的应用。对数据量特别大的应用，在应用架构设计中通常使用数据库底层排序以及排序后的分页访问，这样可提升整个系统的效率和响应能力。为应对巨大数据量的排序要求，作者正考虑将相关算法实现为存储过程，以便于在数据库内实现中英文字串分别按读音及字典序排序。

参考文献：

[1] CHENG Xiaojin, XU Xiuhua. On garbled solution in programming development [J]. Journal of Beijing Institute of Graphic Communication, 2011, 19 (4): 60-62 (in Chinese). [程晓锦，徐秀花. 应用程序开发中的乱码问题 [J]. 北京印刷学院学报，2011，19 (4)：60-62.]

[2] LI Jianghua, ZHAO Guohui. Research on the question of Chinese encoding in Java [J]. Software Guide, 2009, 8 (3): 49-50 (in Chinese). [李江华，赵国辉. Java 中文编码问题研究 [J]. 软件导刊，2009，8 (3)：49-50.]

[3] LIU Huanhuan, LU Feng, ZHAO Yunshan. A quick sort and fuzzy retrieving approach for Chinese text in Java environment [J]. Computer Knowledge and Technology, 2009, 5 (7): 1664-1666 (in Chinese). [刘焕焕，陆锋，赵云山. 一种适合 Java环境的中文快速排序和模糊检索方法 [J]. 电脑知识与技术，2009，5 (7)：1664-1666.]

[4] QIAO Zhiqiang. Obtaining simplified Chinese phonetic characters with SQL server [J]. Computer Development & Applications, 2011, 24 (11): 29-30 (in Chinese). [乔治强. 在 SQL 中实现获取汉字拼音简码 [J]. 电脑开发与应用，2011，24 (11)：29-30.]

[5] LIU Jia, HAN Xiuling. Implementation and improvement on automatic conversion technology of pinyin into Chinese characters [J]. Science Technology and Engineering, 2007, 7 (24): 6348-6352 (in Chinese). [刘佳，韩秀玲. 拼音到汉字自动转换技术的改进与实现 [J]. 科学技术与工程，2007，7 (24)：6348-6352.]

[6] SUN Hongkai, WANG Yanxun. Study on the method of alphabetization and quick search [J]. Microcomputer Information, 2007, 23 (3): 255-257 (in Chinese). [孙宏凯，王彦勋. 中文数据排序与快速检索方法研究 [J]. 微计算机信息，2007，23 (3)：255-257.]

（下转第 2023 页）

动增长的词库构建方式相比，前者切分准确率更高，中文分词效果更好。采用智能词库更新技术的方式，在搜索主题分类问题上，分类准确率更高。搜索主题分类准确率提升将进一步提高用户搜索体验，此问题的研究是感知用户搜索意图的一个重要方面。

参考文献：

[1] Argenton C, Prüfer J. Search engine competition with network externalities [J]. Journal of Competition Law and Economics, 2012, 8 (1): 73-105.

[2] Qi fumin, Xie xiaoyao, Jing fengxuan. Application of improved PSO-LSSVM on network threat detection [J]. Wuhan University Journal of Natural Sciences, 2013, 18 (5): 418-426.

[3] Choi H, Varian H. predicting the present with google trends [J]. Economic Record, 2012, 88 (s1): 2-9.

[4] Carvalho P C K, Fischer J S G, Xu T, et al. Search engine processor: Filtering and organizing peptide spectrum matches [J]. Proteomics, 2012, 12 (7): 944-949.

[5] Pomerantz J. Google Scholar and 100 percent availability of information [J]. Information Technology and Libraries, 2013, 25 (2): 52-56.

[6] DU Jing, XIONG Hailing. Algorithm to recongnize unknown Chinese words based on BBS corpus [J]. Computer Engineering and Design, 2010, 31 (3): 630-633 (in Chinese). [都菁，熊海灵. 基于论坛语料识别中文未登录词的方法 [J]. 计算机工程与设计，2010，31 (3)：630-633.]

[7] ZHANG Qingliang, XU Jian. Research on automatic extraction of web sentiment words [J]. New Technology of Library and Information Serveice, 2011, 21 (10): 24-28 (in Chinese). [张清亮，徐健. 网络情感词自动识别方法研究 [J]. 现代图书情报技术，2011，21 (10)：24-28.]

[8] LIU Fangfang, WANG Jing, SHEN Qiwei. Framework for Chinese word segmentation with dynamic updating dictionary [J]. Computer System & Application, 2013, 22 (3): 100-104 (in Chinese). [刘芳芳，王晶，沈奇威. 一种引入动态词库更新的中文分词架构 [J]. 计算机系统应用，2013，22 (3)：100-104.]

[9] JIANG Yingying, TIAN Feng, WANG Xugang, et al. Adaptive symbol recongnition for sketch based interfaces based on template matching and SVM [J]. Chinese Journal of Computers, 2009, 32 (2): 253-256 (in Chinese). [姜映映，田丰，王绪刚，等. 基于模板匹配和SVM的草图符号自适应识别方法 [J]. 计算机学报，2009，32 (2)：253-256.]

[10] ZHOU Ge. Reverse text frequency based on mutual information on text categorization [J]. Application Research of Computers, 2012, 29 (2): 487-489 (in Chinese). [周戈. 一种基于反向文本频率互信息的文本挖掘算法研究 [J]. 计算机应用研究，2012，29 (2)：487-489.]

[11] ZHENG Gengzhong. Study of automatism word segementation arithmetic's application in intelligent answering system [J]. Computer Engineering and Design, 2007, 28 (9): 2224-2226 (in Chinese). [郑耿忠. 自动分词算法在智能答疑系统中的应用研究 [J]. 计算机工程与设计，2007，28 (9)：2224-2226.]

[12] YAO Xu, GUO Shuni, LI Yonghong, et al. Design of Tibetan word segmentation dictionary with multi-level index [J]. Journal of Computer Apllications, 2009, 29 (6): 178-180 (in Chinese). [姚徐，郭淑妮，李永宏，等. 多级索引的藏语分词词典设计 [J]. 计算机应用，2009，29 (6)：178-180.]

(上接第2012页)

[7] Ken Arnold, James Gosling, David Holmes. The Java programming language [M]. 4th ed. CHEN Haopeng transl. Beijing: Addison Welsley Professional, 2006 (in Chinese). [Ken Arnold, James Gosling, David Holmes. Java程序设计语言 [M]. 4版. 陈昊鹏，译. 北京：人民邮电出版社，2006.]

[8] 911 Chinese-Pingyin query web site [EB/OL]. [2013-07-18]. http://pinyin.911cha.com(in Chinese). [911汉字拼音查询网站. [EB/OL]. [2013-07-18]. http://pinyin.911cha.com.]

[9] Dictionary Editorial Office, Linguisitics Research Institu Chinese Academy of Social Sciences. Xinhua dictionary [M]. 11th ed. Beijing: The Commercial Press, 2011 (in Chinese). [中国社会科学院语言研究所词典编辑室. 新华字典 [M]. 11版. 北京：商务印书馆，2011.]

[10] Dictionary Editorial Office, Linguisitics Research Institute, Chinese Academy of Social Sciences. Modern Chinese Dictionary [M]. 6th ed. Beijing: The Commercial Press, 2011 (in Chinese). [中国社会科学院语言研究所词典编辑室. 现代汉语词典 [M]. 6版. 北京：商务印书馆，2011.]