在嵌入式系统中实现具有联想功能的汉字输入法的研究

李 明[1] 李方军[2] 张为群[3] 邹显春[3] 徐永红[2]

(重庆教育学院信息中心 重庆 400067)[1] (中国工程物理研究院工学院 绵阳 621900)[2]

(西南大学计算机与信息技术学院 重庆 400715)[3]

摘 要 在嵌入式系统中实现拼音输入法可以采用数组和有序树两种方法,其中有序数方法更具有优势。本文采用改进的 Trie 树构造拼音生成树,在 uClinux 上实现了具有联想功能的汉字输入法。

关键词 嵌入式系统,拼音输入法,Trie 树

The Implementation of Chinese Pinyin Input in Embedded Systems

LI Ming[1] LI Fang-Jun[2] ZHANG Wei-Qun[3] ZHOU Xian-Chun[3] XU Yong-Hong[2]

(Network Information Center, Chongqing Educational College, Chongqing 400067)[1]

(Staff College, China Academy of Engineering Physics, Sichuan, Mianyang 621900)[2]

(Faculty of Computer and Information Science, Southwest University, Chongqing 400715)[3]

Abstract The Chinese pinyin input is usually implemented in embedded systems based upon array or ordered tree. We improved the structure of Trie tree, a kind of ordered tree, to build a pinyin tree, and then implement the intelligent Chinese pinyin input under uCLinux.

Keywords Embedded systems, Pinyin input, Trie tree

1 引言

在手持设备和仪器仪表等嵌入式系统中,一般要求实现汉字的输入,开发这样的系统需要对输入法进行研究。输入法就是根据用户的输入查找出相应的汉字供用户选择,它分为两部分,一是运用分析、综合的方法将汉字排序,称为"排",也就是字库和码表的生成;二是运用一定的方法将需要的汉字从字库中取出来,称为"检",其难点在于检索算法的构造[1]。常用的汉字输入分为拼音输入和笔画输入两类,本文研究拼音输入。拼音输入法是一种有重码的输入法,即一个拼音对应多个汉字。采用拼音音序排检汉字的方法,主要有两种,第一是完全按《汉语拼音方案》拼写词连缀的字母顺序排列,例如 mianhua(棉花),这也是我们在 Windows 中常用的全拼、智能拼音采用的方法,它包含汉字、词语、成语,因为在词语、成语中汉字大量重复,字库较大,经过 Windows 上码表逆转化发现,其体积约有 1M 多,这在 PC 机上实现是可以的,但是对于嵌入式系统则过于庞大。第二是以字母排序,辅以音调,这也不适合在嵌入式系统中使用。我们将两种方法结合起来,去掉词语和成语,将汉字按字母排序,如第一个字母相同,则比较第二个,依此类推,若是字母都相同,则按音调排序,这样得到精简后的字库只有几十 k,适合在嵌入式系统中使用。

在普通的 PC 机上,由于系统资源较为充足,输入法的实现主要考虑功能全、易用性,而在嵌入式系统中,资源非常有限,主要考虑算法的高效性。在嵌入式系统中应用的输入法具有如下要求:结构紧凑,便于移植;体积较小,因为嵌入式系统的存储空间有一定限制;检索效率高,因为嵌入式处理器的速度有限,检索效率直接影响用户的使用;最好具有联想功能,加快用户的输入,增加联想功能使得输入法变得庞大,又不利于在嵌入式系统中使用,可以根据实际的情况加以选择。

2 嵌入式系统拼音输入法的实现方法

在嵌入式系统中,构造拼音外码到内码的转换通常有两种方法,一是采用数组,二是采用有序树。

2.1 数组方法

在汉字中拼音共有 300 多个,可以将每个拼音定义为一个数组,数组的成员包括不考虑声调时读音相同的汉字。例如,a 和 ai 的数组定义如下所示:

unsigned char py_a[ ]={"阿啊"};

unsigned char py_ai[ ]={"哎哀唉埃挨皑癌矮蔼艾爱隘碍"};

根据用户的输入在相关的数组中查询,得出相应的汉字供用户选择,其实现方法比较简单,对于要求汉字不多,效率要求不高的场合较为适用。

采用数组的方法虽然实现简单,但具有固有的缺点。首先,该方法不易实现汉字的联想功能。其次,对于每一个拼音相同的汉字归为一组,在程序运行之初便静态分配数组,占用较多的系统资源,在某些存储器不足的嵌入式系统中不能很好地运行。

2.2 键树及其改进

静态数组不能动态释放内存,由数据结构算法可知,其检索效率不高,因此有必要将拼音构成一棵有序树,动态生成与释放,并且在树中的结点域中不包含汉字,节省资源。这里宜采用键树来实现。键树是一种特殊的查找树,其树中每个结点不是通常意义的关键字,而是组成关键字中的一个字符,从根到叶子结点的一条"路径"才对应一个关键字。若关键字是

李 明 副教授,主要研究方向:计算机网络技术、计算机应用技术、现代教育技术。李方军 副教授、博士。张为群 教授。邹显春 副教授,硕士。徐永红 副教授。

数值，则结点中只包含一个数位，若关键字是单词，则结点中只包含一个字符[2]。

设字符集的集合{cai,cao,li,lan,cha,wen,yun,yang,liu,chen}，按首字母将其分解为{cai,cao,cha,chen}，{li,liu,lan}，{wen}，{yang,yun}，对于关键字个数大于一的集合再按第二个字母进行分解为{{cai,cao}，{cha,chen}}，别的集合也按类似的方法分解。显然，按此方法分解的集合很容易生成一棵有序树(图 1)，即同一层的兄弟结点之间所含的字符从左至右有序。

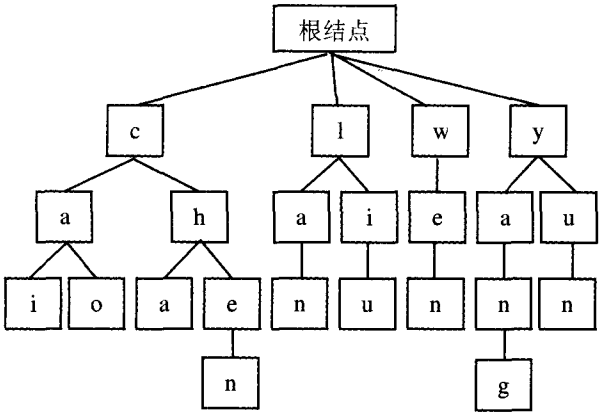

图 1　键树结构

从根结点到某子结点或叶子结点的一条路径构成一个关键字，对于某结点的子结点来说，它是有序的，即从左至右由小到大排列，这样将有利于构造和检索，但是在图 1 中每个结点包含的域是不定的，这在构造时不利于生成拼音树，因此需对此进行改造。

通常键树有两种存储结构。一种是双链树，其分支结点包含三个域，即关键字域、指向第一棵子树根的指针和指向右兄弟的指针。当键树作为“索引”结构时，通常键树中所含关键字的数量较大，此时宜采用多叉链表作为存储结构，用这种存储结构存储的键树称为 Trie 树。这种结构占的空间较大，然而可极大地提高查找速度。若关键字仅由英文字母组成时，树中每个结点可由 27 个指针域组成。双链树用于拼音输入时不足以描述拼音对应的字库信息以及兄弟结点和父结点的关系，而 Trie 树包含 27 个域，其中 26 个对应于 26 个字母，事实上，最长的拼音也只有 6 个字母，即构造的树深度为 6，标准的 Trie 树中，每个结点包含 27 个域，其内存占用空间相当大，因此不适合在拼音输入法中直接使用，需要加以改进，比如将空白的根结点去掉、设置灵活的域的个数等。

下面是改进的拼音树定义，每个结点包含 3 个域和 3 个指向结构体的指针。

```
struct node_char
{
  char key; //该结点的关键字为一字母
  short start;
  //由根结点至该结点的路径表示的拼音对应的汉字在字库的起始位置
  short number; //该拼音相同的汉字的个数
  struct node_char *child_char, *next_char, *parent_char;
  //子结点、右兄弟结点和父结点
}root_char[26];
```

3　拼音输入法在 uClinux[3] 上的实现

3.1　拼音输入法组成

汉字输入的关键是根据用户的输入，从给定的字库中检索出相应的汉字，即由用户的输入得出相应的汉字的位置信息，根据该信息检索相关汉字供用户选择，具体实现分为生成树模块、检索模块、联想模块。

(1)生成树模块

为了实现该树的自动生成，需要构造一张拼音码表，该表主要包含拼音和它对应的汉字的个数。为了自动地生成有序的拼音树，该表中的拼音必须按照以字母从小到大排列，即首先比较第一个字母，第一个字母相同再比较第二个字母，以此类推。生成树模块主要从拼音表中读取拼音对应的汉字信息，采用递归的方法自动实现。构造的树如图 2 所示，为了便于分析，我们将该树倒置，图中只画了一部分，包含的拼音有 a、ai、an、ang、ao、ba 和 da。结点 a 位于第一层，其父结点为空，叶子结点无子结点，在查找过程中，当查找到某结点的子结点为空时，表明为叶子结点，即此次查找结束。

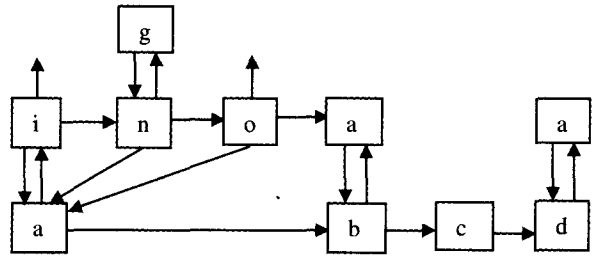

图 2　拼音生成树结构

(2)检索模块

检索模块主要是根据用户的输入在树中查询相应的结点。从根结点出发，沿着与给定值相应的指针逐层向下，若分支结点值和给定的值相等，则停止查找，若不相等，则继续向下查找，直至叶子结点。若叶子结点相应的域值和给定值相等，则查找成功，否则，查找过程失败。当查找成功时，取出相应结点包含该拼音对应的汉字的位置和个数信息，因此很容易以这两个信息作为参数从字库中得到拼音相同的汉字。

(3)联想模块

联想功能是在用户完成一个汉字输入时，提示相关的汉字供用户选择。要实现联想功能，需要重新构造一张表，称为联想码表，表中保存用户选择的汉字在字库中位置信息和由该汉字开头的联想到的汉字在字库中位置信息。将一个汉字对应在字库中的位置信息称为地址，则联想码表即是用户选择的汉字地址和该汉字对应的联想汉字的地址表，通过这张表，可以自动构造一棵数字键树(图 3)。

```
struct node_figure
{
    short figure;// 该结点的关键字，为 0-9 的数字
    short add[9];//联想汉字的地址数组，保存相应汉字的地址信息
    struct node_figure *child_figure, *next_figure, *parent_figure;
}root_figure[10];//树中结点类型只有 10 种
```

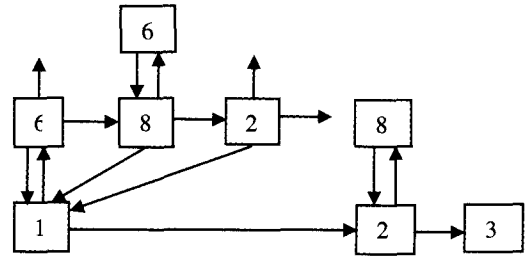

图 3　联想码表生成树结构

在该树中，结点的关键字不再是字母，而是数字。由根结点到分支结点或是叶子结点的路径表示用户选择的汉字的地

址信息(内码),当用户从拼音相同的汉字中选出所需的汉字时,将该字在字库中的地址作为参数传递到联想子程序,并以该信息作为检索的条件,再从根结点到分支结点或是叶子结点进行查询,类似于在生成树中查询,然后得到由该字联想到的汉字的地址信息,再从字库中取出汉字供用户选择。以词语"地球"的输入为例,根据第一个汉字内码和用户选择的汉字在这组汉字中的位置,可以获得用户选择的汉字的内码,如"地"的内码为2080,以该内码为参数,在联想码表生成树中查找该内码的对应汉字的联想汉字,从其取出由"地"联想到的汉字,以一个静态数组记录其内码,"球"的内码为8980,再显示该组汉字,"地"联想的汉字常用的就一个"球"字,若有多个则顺序显示。至此,完成用户对于"地球"的输入。

由于联想码表生成树中,一条路径表示联想汉字的地址,又因为一个汉字在字库中占两个字节,所以,有效的路径其分支结点或是叶子结点都是偶数。

改进的Trie树的查找过程都是从根结点出发,走一条从根到叶子结点的路径,其查找时间依赖于数的深度,在拼音生成树中,树的深度为6,因为最长的拼音由6个字母组成;而在联想码表生成树中,常用汉字有5000多个,根据每个汉字占2个字节,其地址信息需要6个数字表示,树的深度也为6。

3.2 Makefile文件

系统在uClinux下开发所使用的Makefile文件内容为:

```
EXEC=pinyin
OBJS=pinyin.o graphic.o
all:$(EXEC)
$(EXEC):$(OBJS)
    $(CC) $(LDFLAGS)-o $@ $(OBJS) $(LIBM) $(LDLIBS)
    $(LIBGCC)
romfs:
    $(ROMFSINST)/bin/$(EXEC)
    $(ROMFSINST)/pinyin/pymb
    $(ROMFSINST)/pinyin/lxmb
    $(ROMFSINST)/pinyin/ziku
clean:
    -rm -f $(EXEC) *.elf *.gdb *.o
```

执行make romfs时调用"romfs"部分,ROMFSINST是一个宏调用,其功能是将生成的可执行文件复制到目标机上的romfs目录,供打包烧写使用。

当执行make clean时调用该clean部分,清除编译过程中生成的中间代码。

3.3 应注意的几个问题

第一,一个汉字占两个字节,因此,在计算汉字起始位置时必须将前面的汉字个数乘以2。

第二,汉字拼音中没有以i、u、v开头的,在生成树模块中,构造的拼音码表也就没有该内容,在生成树中也就没有相应的结点。在处理用户输入过程中首先加以判断处理,若是以i、u、v开头的输入可以不到树中检索,以节约时间,否则,再从根结点开始检索。

第三,字库的构造可以采用Windows中的码表生成器。选择合适的码表进行逆转换后得到字库,再根据系统实际的需要,对字库进行相应的改造处理。

结论 汉字数组的实现方法简单,采用数组的方法要将汉字数组加载进内存,占用资源较大,适用于汉字不多的场合,比如在手持设备中,要求的汉字不多,或是经常需要的汉字比较固定,可以采用该方法。在数组实现方法中,汉字被以静态数组的形式加载到内存,没有字库,不易实现汉字的联想输入。并且在数组中,依靠下标顺序检索汉字,效率较低。

采用拼音树的方法,查找主要依靠指针传递地址,效率高,并且在程序运行之初,汉字库并不加载到内存,比较节约资源。同时字库的扩充比较方便,只需要对码表进行修改,而不需要修改源代码,便于应用程序的移植。

本文在uClinux环境下开发了拼音输入法,通过运行证明该算法可行,检索速度较高。

参考文献

1 黄俊贵,倪波.汉字与汉字排检方法[M].书目文献出版社,1990

2 严蔚敏,吴伟民.数据结构.清华大学出版社[M],1997

3 http://www.uclinux.org

(上接第210页)

个样本上使用两种方法的错识个数。决策树方法的错识个数明显少,原因是训练充分时它生成的判定分支要比人工预测全面。比较两条曲线,决策树方法的曲线波动比较小,说明其性能比较稳定,即对用户并不敏感,进一步说明了本文采用的方法具有不错的用户适应性。

结论 实验结果说明了本文采用的在线手写数字识别方法是快速高效的,识别的正确率和响应速度都是令人满意的,方法的良好用户适应性也通过实验得到了验证。

下一步的研究工作是如何得到更高的识别率和更好的用户适应性。有两个方面可以完善:第一,优化笔划走势特征的提取算法,更好地捕捉手写数字变体间"形似"的特性。第二,适当增加分类属性特征,以适应更多的手写数字变体。

参考文献

1 Guyon I,Warwick C. Handwriting as computer interface. Survey of the State of the Art in Human Language Technology, NSF. 1995

2 Hansmann U,Merk L,Nicklous M S,et al. Pervasive Computing. Second Edition. Berlin Heidelberg:Springer-Verlag,2001

3 Liu Cheng-Lin, Nakashima K, sako H, et al. Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognition, 2003,36:2271~2285

4 Khorsheed M S. Off-Line Arabic Character Recognition - A Review. Pattern Analysis & Application, 2002,5:31~45

5 孙正兴,彭彬彬,丛兰兰,等,在线草图识别中的用户适应性研究.计算机辅助设计与图形学学报,2004,16(9):1207~1215

6 Vuori V,laaksonen J,Oja E,et al. On-Line Adaptation in Recognition of Handwritten Alphanumeric Characters. In: Fifth International Conference on Document Analysis and Recognition, Bangalore, India, 1999. 792

7 Artieres T, Marchand J-M, Gallinari P, et al. Stroke level modeling of on-line handwriting through multi-modal segmental models. IWFHR, 2000

8 Mitchell T M. Machine Learning. In: The McGraw-Hill Companies. Inc, 1997,39