

判断唯一可译码

ismdeep

2013-10-17 10:56:58

1 唯一可译码

任意有限长的码元序列，只能被唯一地分割成一个个的码字，便称为唯一可译码。

2 判断唯一可译码

将码C中所有可能的尾随后缀组成一个集合F，当且仅当集合F中没有包含任一码字，则可判断此码C为唯一可译变长码。如何构成集合F，可以如下进行。

首先，观察码C中最短的码字是否是其他码字的前缀。若是，将其所有的可能的尾随后缀排列出。而这些尾随后缀又可能是某些码字的前缀，再将由这些尾随后缀产生的新的尾随后缀列出。

然后再观察这些新的尾随后缀是否是某些码字的前缀，再将产生的后缀列出。

依次下去，直至没有一个尾随后缀是码字的前缀或没有新的尾随后缀产生为止。这样，首先获得由最短的码字能引起的所有尾随后缀。接着，按照上述步骤将次短的码字、...等等，所有码字可能产生的尾随后缀全部列出。

由此，得到由码C的所有可能的尾随后缀组成的集合F。

3 实现代码

详细代码在src文件夹中。或者访问网址：

短网址：<http://t.cn/zRVLbj0>

原网址：https://github.com/ismdeep/public_documents/blob/master/%E4%BF%A1%E6%81%AF%E5%AE%89%E5%85%A811%E7%8F%AD/%E4%BF%A1%E6%81%AF%E8%AE%BA/%E5%88%A4%E6%96%AD%E5%94%AF%E4%B8%80%E5%8F%AF%E8%AF%91%E7%A0%81/src/main.cpp

```
1 #include <iostream>
2 #include <stdlib.h>
3 #include <string.h>
4
5 using namespace std;
6
7 struct strings
8 {
9     char *string;
10    struct strings *next;
11 };
12
13
14 struct strings Fstr, *Fh, *FP;
15 void outputstr(strings *str)
16 {
17     do
18     {
19         cout<<str->string<<endl;
20         str = str->next;
21     } while (str);
22     cout<<endl;
23 }
```

```

24 inline int MIN(int a, int b)
25 { return a>b?b:a; }
26 inline int MAX(int a, int b)
27 { return a>b?a:b; }
28 #define length_a (strlen(CP))
29 #define length_b (strlen(tempPtr))
30 //判断一个码是否在一个码集合中, 在则返回, 不在返回0
31 int comparing(strings *st_string, char *code)
32 {
33     while(st_string->next)
34     {
35         st_string=st_string->next;
36         if(!strcmp(st_string->string, code))
37             return 0;
38     }
39     return 1;
40 }
41
42 //判断两个码字是否一个是另一个的前缀, 如果是则生成后缀码
43 void houzhui(char *CP, char *tempPtr)
44 {
45     if (!strcmp(CP, tempPtr))
46     {
47         cout<<"集合和集合中有相同码字CF:"<<endl
48             <<CP<<endl
49             <<"不是唯一可译码码组!"<<endl;
50         exit(1);
51     }
52     if (!strncmp(CP, tempPtr, MIN(length_a, length_b)))
53     {
54         struct strings *cp_temp;
55         cp_temp=new (struct strings);
56         cp_temp->next=NULL;
57         cp_temp->string=new char[abs(length_a-length_b)+1];
58         char *longstr;
59         longstr=(length_a>length_b ? CP : tempPtr);//将
长度长的码赋给longstr
60
61         //取出后缀
62         for (int k=MIN(length_a, length_b); k<MAX(length_a, length_b); k++)
63             cp_temp->string[k - MIN(length_a, length_b)]=longstr[k - MIN(length_a, length_b)];
64         cp_temp->string[abs(length_a-length_b)]=NULL;
65         //判断新生成的后缀码是否已在集合里, 不在则加入集合
66         if (comparing(Fh, cp_temp->string))
67         {
68             FP->next=cp_temp;
69             FP=FP->next;
70         }

```

```

71         }
72     }
73
74
75     int main()
76     {
77         cout<<"\t\唯一可译码的判断t!\n"<<endl;
78         struct strings Cstr, *Ch, *CP,*tempPtr;
79         Ch=&Cstr;
80         CP=Ch;
81         Fh=&Fstr;
82         FP=Fh;
83         char c []="C_:";
84         Ch->string=new char [ strlen (c) ];
85         strcpy (Ch->string , c);
86         Ch->next=NULL;
87         char f []="F_:";
88         Fh->string=new char [ strlen (f) ];
89         strcpy (Fh->string , f);
90         Fh->next=NULL;
91         //输入待检测码的个数
92         int Cnum;
93         cout<<"输入待检测码的个数:";
94         cin>>Cnum;
95         cout<<"输入待检测码"<<endl;
96         for (int i=0; i<Cnum; i++)
97         {
98             cout<<i+1<<"_:";
99             char tempstr [10];
100             cin>>tempstr;
101             CP->next=new (struct strings);
102             CP=CP->next;
103             CP->string=new char [ strlen (tempstr) ] ;
104             strcpy (CP->string , tempstr);
105             CP->next = NULL;
106         }
107         outputstr (Ch);
108         CP=Ch;
109         while (CP->next->next)
110         {
111             CP=CP->next;
112             tempPtr=CP;
113             do
114             {
115                 tempPtr=tempPtr->next;
116                 houzhui (CP->string ,tempPtr->string );
117             } while (tempPtr->next);
118         }

```

```

119     outputstr(Fh);
120     struct strings *Fbegin,*Fend;
121     Fend=Fh;
122     while(1)
123     {
124         if(Fend == FP)
125         {
126             cout<<"是唯一可译码码组!"<<endl;
127             exit(1);
128         }
129         Fbegin=Fend;
130         Fend=FP;
131         CP=Ch;
132         while(CP->next)
133         {
134             CP=CP->next;
135             tempPtr=Fbegin;
136             for(;;)
137             {
138                 tempPtr=tempPtr->next;
139                 houzhui(CP->string,tempPtr->string);
140                 if(tempPtr == Fend)
141                     break;
142             }
143         }
144         outputstr(Fh); //输出集合中全部元素F
145     }
146     return 0;
147 }

```