# 哈夫曼译码器

【需求分析】

编写一个系统对待传数据进行预先编码，在接收传来的数据时进行译码。

对空格符及大写字母A至Z进行预先编码，在接收输入的报文时将其按照预先编码进行译码。

【概要设计】

构建结构体数组存储哈夫曼树的每个结点

typedef struct nodelist{

int f;

char data;

int lchild;

int rchild;

}tree;

tree hfm[ ];

读入数据并建立哈夫曼树：

void Initialization(int a[],char b[],int n,tree hfm[])

{

int pot[100],i,m,mm,s=n,min,mmin,t;

for (i=0;i<100;i++) pot[i]=0;

for (i=0;i<n;i++) {hfm[i].data=b[i]; hfm[i].f=a[i]; pot[i]=1; hfm[i].lchild=-1; hfm[i].rchild=-1;}

while (s<2\*n-1)

{

min=1000;mmin=1001;

for (i=0;i<s;i++)

{

if (pot[i]!=0&&hfm[i].f<mmin) {mmin=hfm[i].f; mm=i;}

if (mmin<min) {t=min; min=mmin; mmin=t; t=mm; mm=m; m=t;}

}

hfm[s].lchild=mm; hfm[s].rchild=m;

hfm[s].f=min+mmin;

pot[m]=0; pot[mm]=0; pot[s]=1; s++; }

return;

}

利用建立好的哈夫曼树进行编码：

void makeit(int a[],char c[][27],int q,tree hfm[]){

int r;

if (hfm[q].lchild>=0) {r=hfm[q].lchild; strcpy(c[r],c[q]); c[r][strlen(c[r])]='0'; makeit(a,c,r,hfm);}

if (hfm[q].rchild>=0) {r=hfm[q].rchild; strcpy(c[r],c[q]); c[r][strlen(c[r])]='1'; makeit(a,c,r,hfm);}

}

译码函数：

char translate(char s[],tree hfm[],int q)

{

int i=0,j=q;

while (i<strlen(s))

{

if (s[i]=='0') j=hfm[j].lchild;

else j=hfm[j].rchild;

i++;

}

return hfm[j].data;

}

输入原始数据并进行译码和输出：

tree hfm[100];

intn=27,a[100]={186,64,13,22,32,103,21,15,47,57,1,5,32,20,57,63,15,1,48,51,80,23,8,18,1,16,1},i,j,q;

char b[27]={' ','A','B','C','D','E','F','G','H','I','G','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z'};

char c[100][27];

char s[100]={"THIS PROGRAM IS MY FAVORITE"},secret[100][27]; ，secret

Initialization(a,b,n,hfm);

q=2\*n-2;

for (i=0;i<q;i++)

for (j=0;j<27;j++)

c[i][j]='\0';

makeit(a,c,q,hfm);

for (i=0;i<strlen(s);i++)

{

if (s[i]==' ') strcpy(secret[i],c[0]);

else strcpy(secret[i],c[s[i]-'A'+1]);

puts(secret[i]);

}

for (i=0;i<strlen(s);i++) printf("%c",translate(secret[i],hfm,q));

}

【测试结果】

输入：THIS PROGRAM IS MY FAVORITE

输出：

1110

1001

1100

000

011101

1101

0110

011110

1101

0101

001101

000

1001

1100

000

001101

011100

000

001100

0101

0011111

0110

1101

1001

0010

101

输入：HELLO WORLD

输出：

1110

101

01000

01000

0110

000

001110

0110

1101

01000

01001

【附录】