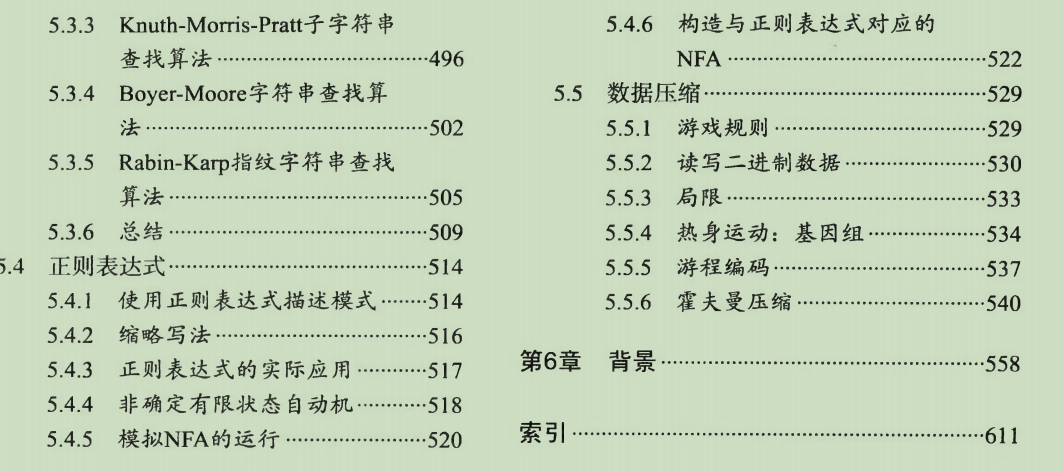
名词解释

1.比特流这个术语表示比特的序列

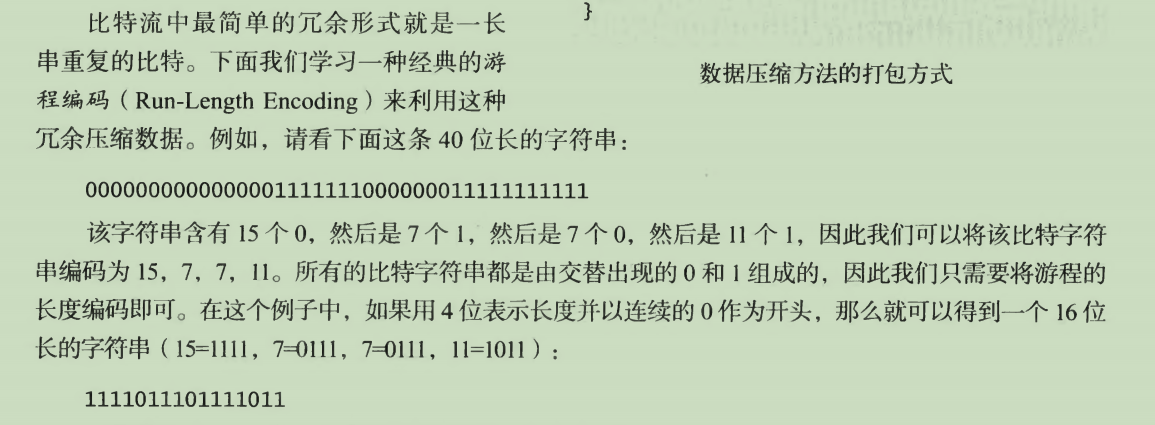
2.字节流这个术语表示可以看作固定大小的字节序列的比特序列



双位编码

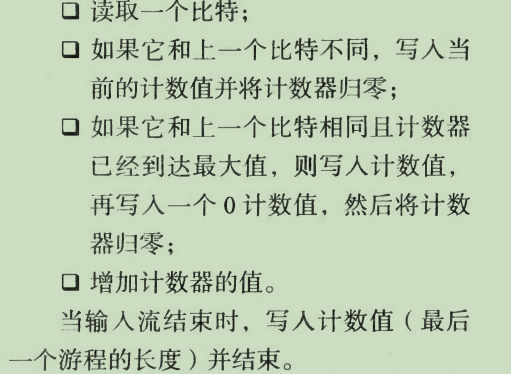
基因组只有四个字符表示，可以将数据存储到两个比特中。一个8字符表示四个基因字符

**游程编码**

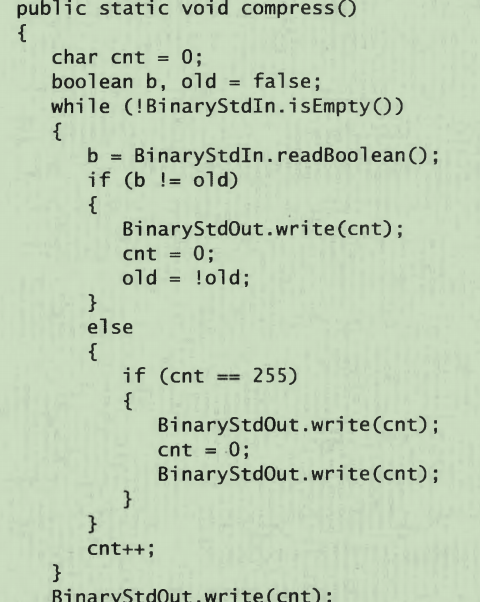


通常用于图片数据压缩，图片数据出现大量重复的0和1.

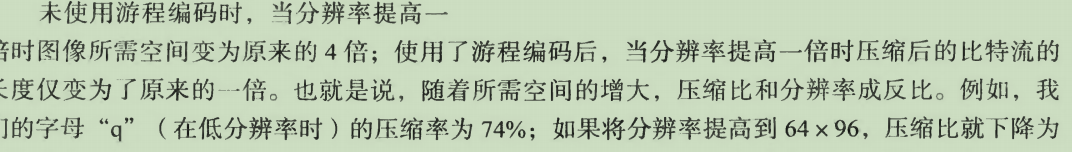
**代码**



**compress用于数据压缩**

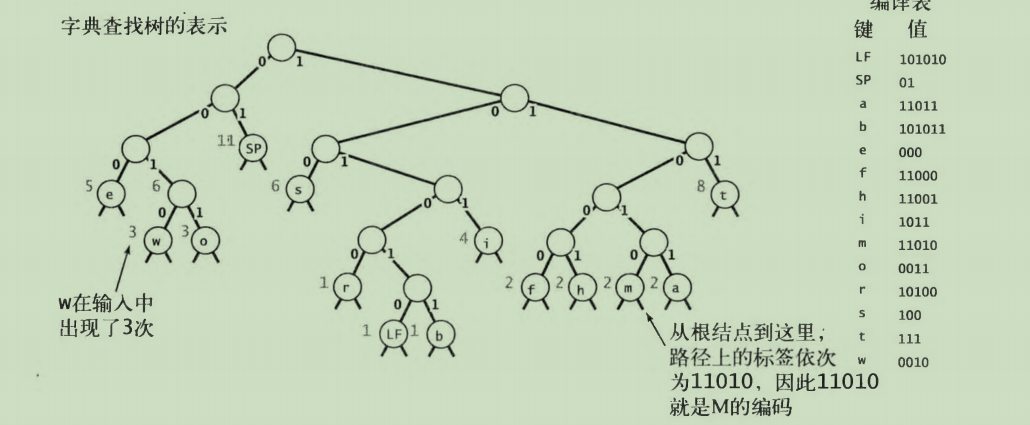


**优势领域**



**霍夫曼编码**

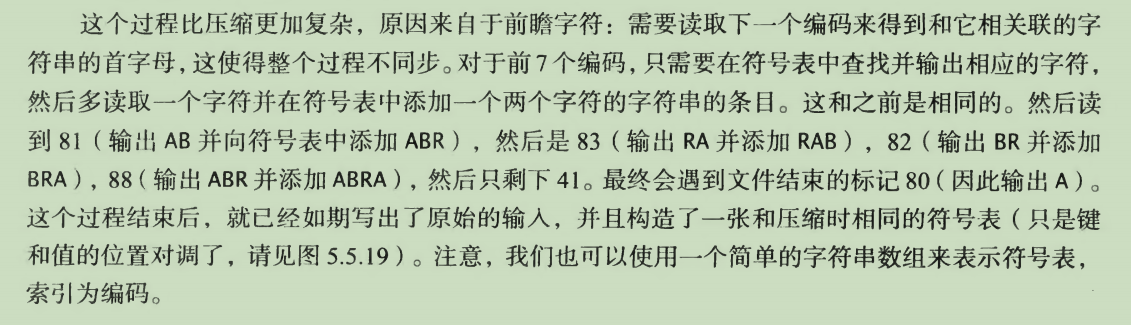
根据字符出现的频率来构造一棵树。适用于各种文本



、

**LZW压缩编码**

**编码原理**

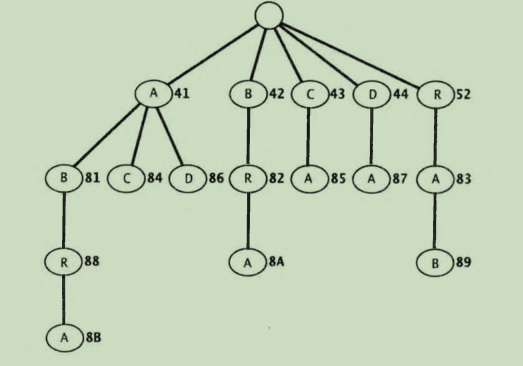


根据普通ascll编码图表，我们将字符a，b，c，1,2,3等这种单个字符的我们直接获取的，限制在0到80范围，80同时代表文本结束，对于像下面一张图片中AB，ABCD等这种，我们从81开始不断地增大，81代表AB，一个8位的比特的字节代表16位A,B两个字节，他的数据压缩就体现在这里。

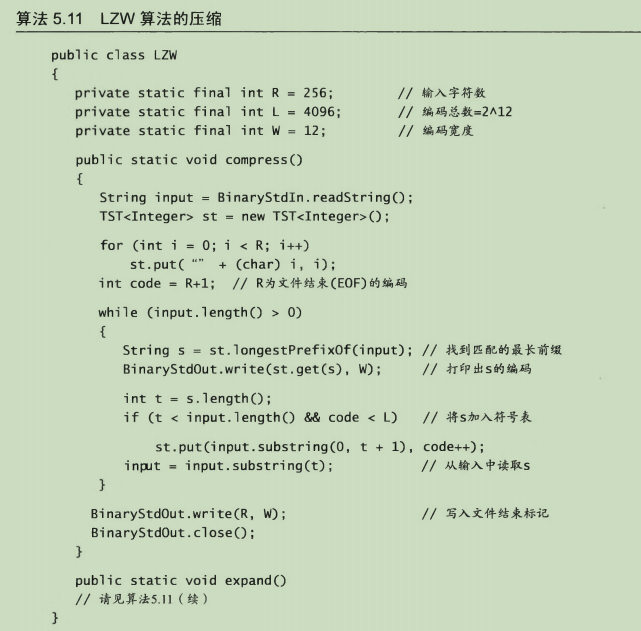
**编码**



压缩过程解释，存在两个索引，第一个前一个字符串索引M和后一个字符串N，对于第一个字符A他的值是41，小于80，所以M+N看是不是已经写入我们的编译表中了，M+N等于AB没有写入编译表所以写入AB 和对应的81，对于后一个字符串N来说每次只要往后面移动一位就可以了，对于M来说就需要找出最长前缀（出现在我们书写的编译表中最长字符串），比如第8步，AB已经出现在编译表中就需要用M来指向AB，不会指向A，如果指向A，下一个为B，又是AB，编译表已经存储过就显得没有任何意义，如何寻找这个最长前缀呢？（下面树解释）

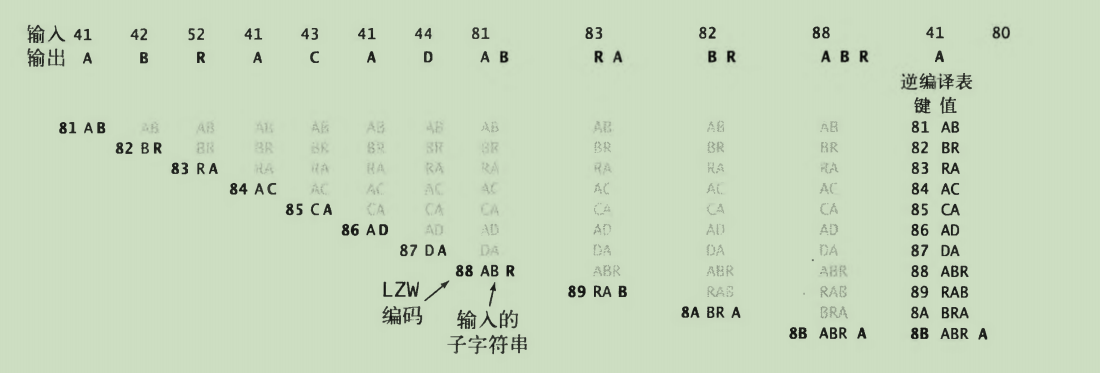


**代码**



**解码过程expand（）**

比压缩过程更加复杂，过程相似，根据小于80的ascell表基础，重新编译出AB等等。



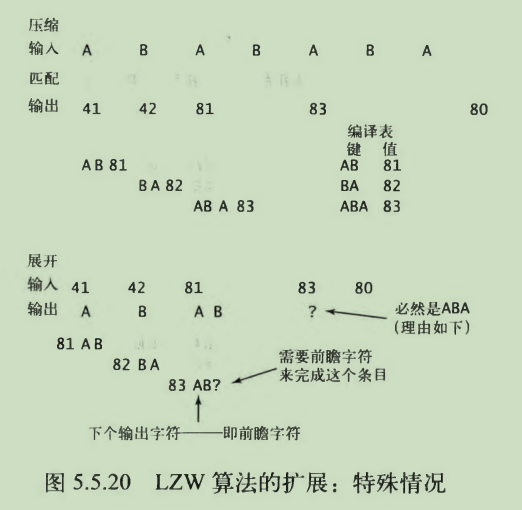
输入

clipboard.png

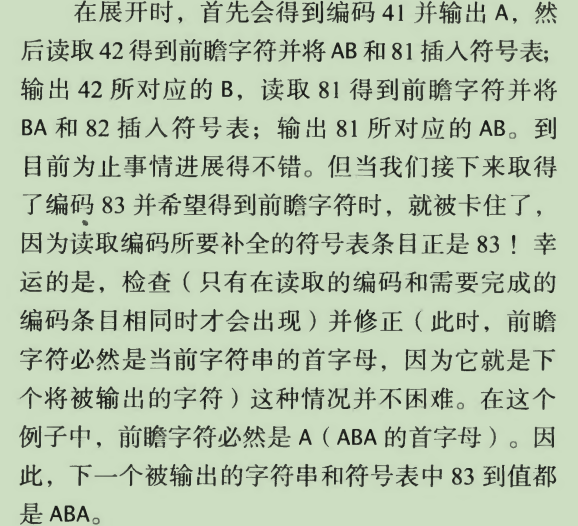
解释41为A （M指向他） 看下一位42为B （N指向他），M字符串+N字符串的第一位，也就是AB 添加81到字符表中（原理如此），这样后面出现81也就可以解释出来，

如果是第8步，M（81）指向AB，N（83）指向RA，如何处理，M(AB)+N的第一个字符（R）存入到字符表添加89，基本原理可以参考上面的编码过程

**特殊情况**



第三步出现问题，我们可以解释出81为AB，但是按照我们上面说的原理，需要解释出83对应字符串取得第一位字符，把两者结合添加到字符表中，83是此时我们需要到字符表中的，我们还不知道83是多少，就要获得83对应的第一个字符，自相矛盾，实际上83第一个字符就是81的第一个字符，如果不是就不符合，你可以自己代进去试一试。



**代码**

