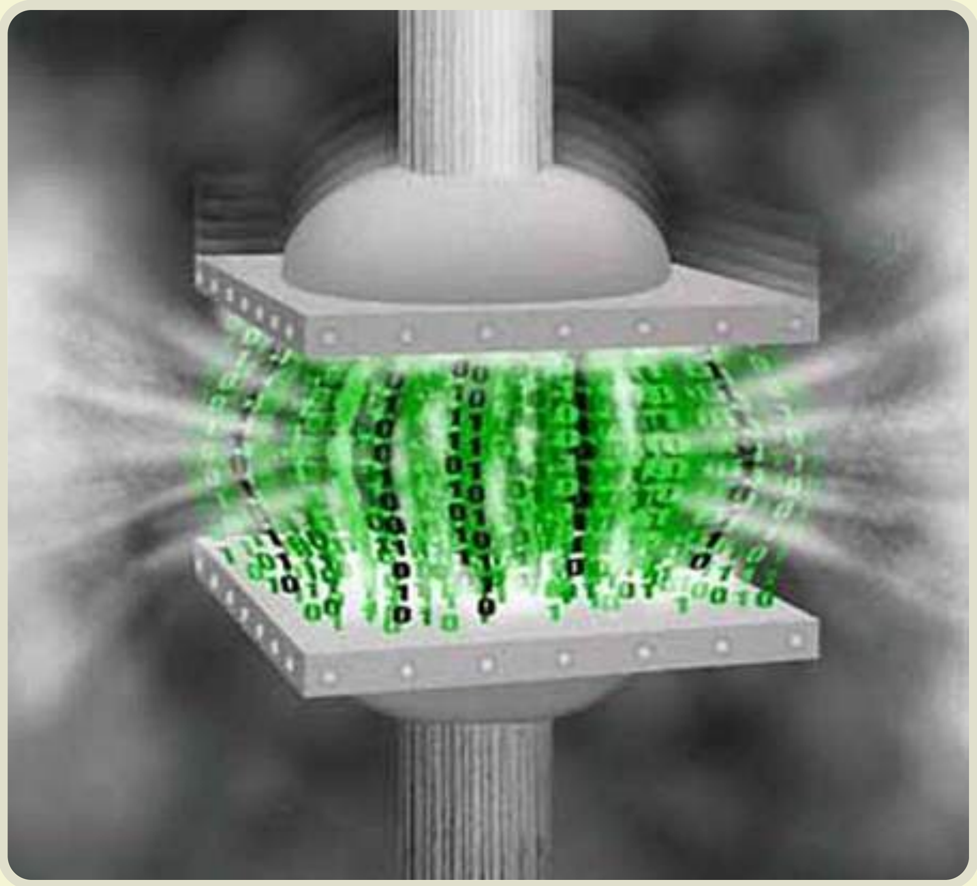
数据压缩与信息熵

1992年，美国佐治亚州的WEB Technology公司，宣布做出了重大的技术突破。

该公司的DataFiles/16软件，号称可以将任意大于64KB的文件，压缩为原始大小的16分之一。业界议论纷纷，如果消息属实，无异于压缩技术的革命。



许多专家还没有看到软件，就断言这是不可能的。因为根据压缩原理，你不可能将**任意文件**压缩到16分之一。事实上，有一些文件是无法压缩的，哪怕一个二进制位，都压缩不掉。

后来，事实果然如此，这款软件从来没有正式发布。没过几年，就连WEB Technology公司都消失了。

那么，为何不是所有的文件都可以被压缩？是否存在一个压缩极限呢，也就是说，到了一定大小，就没法再压缩了？

一、压缩的有限性

首先，回答第一个问题：为什么WEB Technology公司的发明不可能是真的。

反证法可以轻易地证明这一点。假定任何文件都可以压缩到n个二进制位（bit）以内，那么最多有2^n种不同的压缩结果。也就是说，如果有2^n+1个文件，必然至少有两个文件会产生同样的压缩结果。这意味着，这两个文件不可能无损地还原（解压缩）。因此，得到证明，并非所有文件都可以压缩到n个二进制位以下。

很自然地，下一个问题就是，这个n到底是多少?

二、压缩原理

要回答一个文件最小可以压缩到多少，必须要知道压缩的原理。

压缩原理其实很简单，就是找出那些重复出现的字符串，然后用更短的符号代替，从而达到缩短字符串的目的。比如，有一篇文章大量使用"中华人民共和国"这个词语，我们用"中国"代替，就缩短了5个字符，如果用"华"代替，就缩短了6个字符。事实上，只要保证对应关系，可以用任意字符代替那些重复出现的字符串。

本质上，所谓"压缩"就是找出文件内容的概率分布，将那些出现概率高的部分代替成更短的形式。所以，内容越是重复的文件，就可以压缩地越小。比如，"ABABABABABABAB"可以压缩成"7AB"。

相应地，如果内容毫无重复，就很难压缩。极端情况就是，遇到那些均匀分布的随机字符串，往往连一个字符都压缩不了。比如，任意排列的10个阿拉伯数字（5271839406），就是无法压缩的；再比如，无理数（比如π）也很难压缩。

压缩就是一个消除冗余的过程，相当于用一种更精简的形式，表达相同的内容。可以想象，压缩过一次以后，文件中的重复字符串将大幅减少。好的压缩算法，可以将冗余降到最低，以至于再也没有办法进一步压缩。所以，压缩已经压缩过的文件（递归压缩），通常是没有意义的。

三、压缩的极限

知道了压缩原理之后，就可以计算压缩的极限了。

上一节说过，压缩可以分解成两个步骤。第一步是得到文件内容的概率分布，哪些部分出现的次数多，哪些部分出现的次数少；第二步是对文件进行编码，用较短的符号替代那些重复出现的部分。

第一步的概率分布一般是确定的，现在就来考虑第二步，怎样找到最短的符号作为替代符。

如果文件内容只有两种情况（比如扔硬币的结果），那么只要一个二进制位就够了，1表示正面，0表示表示负面。如果文件内容包含三种情况（比如球赛的结果），那么最少需要两个二进制位。如果文件内容包含六种情况（比如扔筛子的结果），那么最少需要三个二进制位。

一般来说，在均匀分布的情况下，假定一个字符（或字符串）在文件中出现的概率是p，那么在这个位置上最多可能出现1/p种情况。需要$\log_2(1/p)$个二进制位表示替代符号。

这个结论可以推广到一般情况。假定文件有n个部分组成，每个部分的内容在文件中的出现概率分别为p_1、p_2、...p_n。那么，替代符号占据的二进制最少为下面这个式子。

$$\log_2(1/p_1) + \log_2(1/p_2) + ... + \log_2(1/p_n)$$

$$= \sum \log_2(1/p_n)$$

这可以被看作一个文件的压缩极限。

四、信息熵的公式

上一节的公式给出了文件压缩的极限。对于n相等的两个文件，概率p决定了这个式子的大小。p越大，表明文件内容越有规律，压缩后的体积就越小；p越

小，表明文件内容越随机，压缩后的体积就越大。

为了便于文件之间的比较，将上式除以n，可以得到平均每个符号所占用的二进制位。

$$\sum \log_2(1/p_n) \ / \ n$$

$$= \log_2(1/p_1)/n + \log_2(1/p_2)/n + \ldots + \log_2(1/p_n)/n$$

由于p是根据频率统计得到的，因此上面的公式等价于下面的形式。

$$p_1*\log_2(1/p_1) + p_2*\log_2(1/p_2) + \ldots + p_n*\log_2(1/p_n)$$

$$= \sum p_n*\log_2(1/p_n)$$

$$= E(\log_2(1/p))$$

上面式子中最后的E，表示数学期望。可以理解成，每个符号所占用的二进制位，等于概率倒数的对数的数学期望。

下面是一个例子。假定有两个文件都包含1024个符号，在ASCII码的情况下，它们的长度是相等的，都是1KB。甲文件的内容50%是a，30%b，20%是c，则平均每个符号要占用1.49个二进制位。

$$0.5*\log_2(1/0.5) + 0.3*\log_2(1/0.3) + 0.2*\log_2(1/0.2)$$

$$= 1.49$$

既然每个符号要占用1.49个二进制位，那么压缩1024个符号，理论上最少需要1526个二进制位，约0.186KB，相当于压缩掉了81%的体积。

乙文件的内容10%是a，10%是b，......，10%是j，则平均每个符号要占用3.32个二进制位。

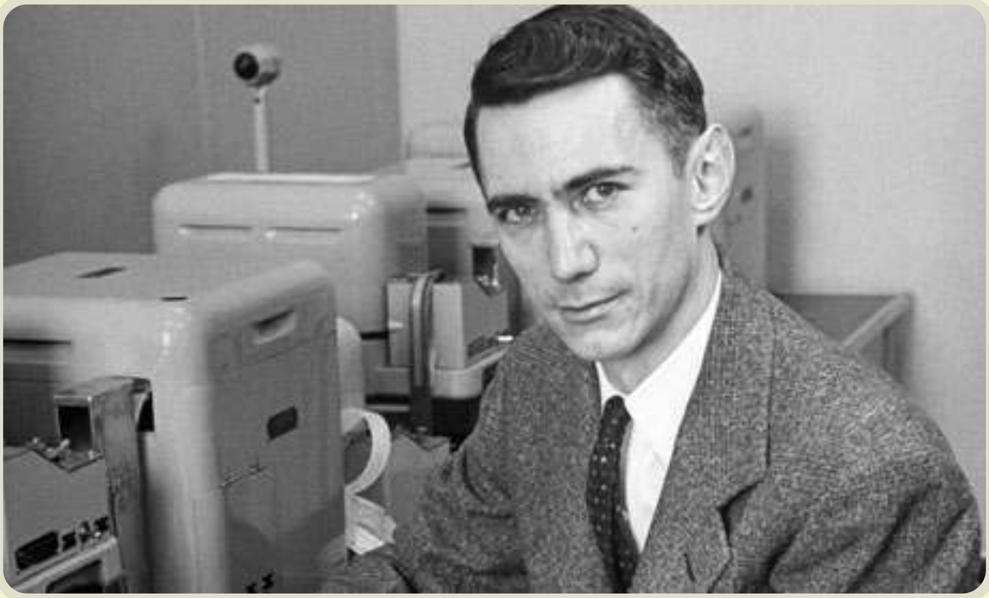
```
0.1*log2(1/0.1)*10

= 3.32
```

既然每个符号要占用3.32个二进制位，那么压缩1024个符号，理论上最少需要3400个二进制位，约0.415KB，相当于压缩掉了58%的体积。

对比上面两个算式，可以看到文件内容越是分散（随机），所需要的二进制位就越长。所以，这个值可以用来衡量文件内容的随机性（又称不确定性）。这就叫做信息熵（information entropy）。

它是1948年由美国数学家克劳德·香农（Claude Shannon）在经典论文《通信的数学理论》中，首先提出的。



五、信息熵的含义

想要理解信息熵这个概念，有几点需要注意。

（1）**信息熵只反映内容的随机性，与内容本身无关。**不管是什么样内容的文件，只要服从同样的概率分布，就会计算得到同样的信息熵。

（2）**信息熵越大，表示占用的二进制位越长，因此就可以表达更多的符号。**

所以，人们有时也说，信息熵越大，表示信息量越大。不过，由于第一点的原因，这种说法很容易产生误导。较大的信息熵，只表示可能出现的符号较多，并不意味着你可以从中得到更多的信息。

（3）信息熵与热力学的熵，基本无关。这两个熵不是同一件事，信息熵表示无序的信息，热力学的熵表示无序的能量（参见我写的《熵的社会学意义》）。

（文章完）