Lempel-Ziv-Welch压缩数据的误码纠正

王 刚[①] 靳彦青[②] 彭 华[*①] 张光伟[①]

[①](中国人民解放军战略支援部队信息工程大学 郑州 450000)

[②](国家数字交换系统工程技术研究中心 郑州 450002)

摘 要：无损数据压缩系统在通信传输过程中容易出现错误，会导致码表和重构数据出错并引发误码扩散，影响其在文件系统和无线通信中的应用。针对在通用编码领域广泛使用的无损数据压缩算法LZW，该文分析并利用LZW压缩数据的冗余，通过选取部分编码码字并动态调整其对应的被压缩符号串的长度来携带校验码，提出了具有误码纠正能力的无损数据压缩方法CLZW。该方法不用额外添加数据，也不改变数据规格和编码规则，与标准LZW算法兼容。实验结果表明，用该方法压缩的文件仍然能用标准LZW解码器解压，且该方法可以对LZW压缩数据的误码进行有效纠正。

关键词：Lempel-Ziv-Welch算法；数据压缩；误码纠正

中图分类号：TP911.21 **文献标识码**：A **文章编号**：1009-5896(2020)06-1436-08

DOI: 10.11999/JEIT190520

Error Correction of Lempel-Ziv-Welch Compressed Data

WANG Gang[①] JIN Yanqing[②] PENG Hua[①] ZHANG Guangwei[①]

[①](*PLA Strategic Support Force Information Engineering University*, *Zhengzhou* 450000, *China*)

[②](*National Digital Switching System Engineering & Technology Research Center*, *Zhengzhou* 450002, *China*)

Abstract: Lossless data compression system is prone to bit error and causes error spread during communication transmission, which affects its application to file system and wireless communication. For the lossless data compression algorithm Lempel-Ziv-Welch (LZW), which is widely used in the field of general coding, analyzes and utilizes the redundancy of LZW compressed data, carries the check code by selecting part of the codeword and dynamically adjusting the length of its corresponding compressed string. A lossless data compression method Carrier-LZW(CLZW) with error correction capability is proposed. This method does not need additional data, does not change the data specification and coding rules, and is compatible with the standard LZW algorithm. The experimental results show that the file compressed by this method can still be decompressed by the standard LZW decoder. In the range of error correction capability, the method can effectively correct the error of LZW compressed data.

Key words: Lempel-Ziv-Welch(LZW) algorithm; Data compression; Error correction

1 引言

信源编码技术广泛应用于各种通信系统，其目的是用尽可能少的比特描述信源，所以也叫做数据压缩。来自网络和传感器不断增长的大量数据，需要高速有效的进行传输和存储[1–4]。为此，学术界对数据压缩领域一直在持续进行研究。这些数据压缩技术和方法可划分为两大类，分别是无损压缩和有损压缩[5]。有损压缩舍弃一些人体感官不明显、不影响对数据内容理解的信息，通常应用于图像、视频和音频等的压缩[6–8]。无损压缩主要用于文本、程序和多媒体信息的压缩，数据中不允许有任何信息的损失[9,10]。

Abraham Lempel和Jacob Ziv提出的Lempel-Ziv 77(LZ77)和Lempel-Ziv 78(LZ78)字典编码算法，成为许多压缩工具、压缩程序和数据格式的核心算法。在LZ78的基础上，Welch[11]提出了LZW算法，主要用于传感器数据、生物信息学分类、医学图像数据及水印等信息的无损压缩。

在实际应用时，压缩数据在传输过程中由于噪声和干扰的影响会产生误码，存储介质的损坏和污

收稿日期：2019-07-11；改回日期：2020-03-25；网络出版：2020-03-27

*通信作者： 彭华 phzttyw@126.com

基金项目：国家自然科学基金(61501516, 61572518)

Foundation Items: The National Natural Science Foundation of China (61501516, 61572518)

染会在数据交互等操作时出现问题，还有其它各种无法预测的情况都会导致压缩数据出现错误。由于后续解压过程需依靠前面的解压结果，所以损坏的压缩数据分组会导致其后未损坏压缩数据分组出现解压错误。因此，误码对压缩数据的影响尤其严重，一处错误就会使得整个文件无法恢复[12]，造成信息损失。

无损压缩数据的误码纠正是为了修复从无线信道中获取的损坏压缩数据。在非协作通信中拦截的压缩数据若有错误，就必须使用误码纠正技术进行修复，由于在这种通信模式下无法重新传输受损的压缩数据，所以需从信源层面解决压缩数据的误码纠正难题。

由于信源经过压缩后消除了大量相关性，压缩数据中剩余的冗余极少[13,14]，目前提升Lempel-Ziv (LZ)系列压缩方法纠错能力的研究，主要通过增加校验位对压缩数据进行保护。Wang等人[12]根据信源编码准则和语法规范对信源先验信息进行建模，并据此判断误码在编码数据中所在的区间范围，采用遍历验证法对错误进行修正。Frenkel等人[15]研究了LZW算法的编码规则，使用指数衰减(exponential decay)作为管理和删除LZW字典中不常用词的工具，改进了码字构造及码表优化方法，能对压缩数据进行有限的保护。文献[16]讨论了输入误码对译码字典构造和解压数据重构的影响，并给出一种针对文本压缩数据的前向反馈检错思路。Klein等人[17]将字典编码规则中的多种模式转换为统一的特定编码结构并插入标志位，可以在发生错误的情况下根据设定的结构提供错误检测能力。Zhang等人[18]针对可变长度信源编码，分析了码字不同部分出错时对数据的影响，通过对压缩数据分组并增设校验位，提出了一种联合信源信道编码算法来保护压缩数据。Kwon等人[19]根据字典编码的压缩机制设计了基于规则模式的误码检测算法，虽然不使用额外的校验位，但没有提出可行的纠正方案。Kitakami等人[20]提出对压缩数据中的错误敏感部分采用unary编码，然后把该部分数据复制到压缩数据的开头并添加同步序列。通过搜索同步序列，可以检测到错误敏感部分的误码，并利用该部分的副本恢复压缩数据。Pereira等人[21]针对字典编码数据引入一种新的数据构造方法，通过使用纠错码进行错误检测。上述方法都需要在压缩数据中集成额外的校验位，没有利用压缩数据的剩余冗余，因此会影响压缩性能。而且这些方法改变了数据结构和编码规范，从而无法兼容标准算法。

为了有效解决LZW压缩数据的误码纠正问题，针对现有研究成果需要额外增加校验位且无法兼容标准算法的不足，本文提出一种与标准算法通用的无损数据压缩方法。该方法依据校验信息挑选部分码字，调整其对应的被压缩符号的数量，基于自适应的码字模式传递校验位。该方法在具有误码纠正能力的同时，仍然可以使用标准LZW解码器解压，并且不影响压缩性能。

2　Carrier-LZW(CLZW)算法

标准LZW算法[11]的编码操作分3步：(1)短语提取——LZW将给定的输入文本解析为子符号串，称为短语。为了提取下一个短语X, LZW扫描剩余的输入，直到找到从当前位置开始的最长符号串，该符号串与之前确定的短语Y完全匹配。然后在输入中附加下一个符号a创建新词组，即X=Ya。如果没有找到匹配项，则新短语只包含下一个符号。(2)字典更新——LZW将每个短语作为单独条目形成字典，字典以树形图的形式实现。符号集中的每个符号作为长度为1的短语输入到字典中，并且按顺序依次将整数分配给短语作为其索引。为了表示X，将一个新节点作为子节点添加到表示短语Y的节点，并将下一个整数指定为X的索引。(3)短语编码——为了对输入数据进行编码，LZW用其父节点Y的索引i_Y来编码短语X。由于X由父节点Y表示，所以X的最后一个符号a作为下一个短语的一部分进行编码。

LZW通过贪婪算法寻找最长匹配符号串进行编码[22]，字典中的所有条目具有唯一性[23]，所以压缩数据中没有足够的冗余空间携带校验位。为了与标准算法兼容且不影响压缩性能，就不能采用在压缩数据中额外添加校验码的方法，必须考虑利用压缩冗余来携带校验位。所以，设计并通过调整压缩数据中部分码字对应的符号串，产生编码冗余传递校验位，实现LZW压缩数据的误码纠正，这正是本文提出的Carrier-LZW(CLZW)算法的思路。

有限符号集中，n为原始数据T的长度，T' 表示T的LZW压缩数据，压缩数据T' 中传递的校验码记为M。CLZW算法中调整了符号串长度的LZW短语叫做非贪婪短语(Non Greedy Phrase, NGP)。CLZW算法有两个参数K和L, K表示通过NGP的密度传递的比特数，L表示通过NGP的长度传递的比特数。

对检验码M进行分组，每组分别由K bit和L bit数据组成，其值用k和l表示($1\le k\le 2^K$, $1\le l\le 2^L$)。按照LZW算法进行压缩时，统计符号串的长度大于2^L的码字数量，当达到k时就将该码字对应的短

语去掉l个符号后再次进行压缩。解压时，读入码字并重构短语，解码器需要判断每个短语的类型，根据设计的携带方法能提取出NGP中传递的$(K+L)$ bit消息，并解压重构出原始数据。

压缩过程中，标准LZW算法在字典中寻找待编码符号串的最长匹配作为编码结果，然后在最长匹配的后面增添1个符号构成新的短语并加入到字典中，此时字典中的每个短语都是唯一的。当使用CLZW算法在字典中寻找匹配前缀时，由于已经缩短了NGP所对应符号串的长度，因此不是最长匹配。当在匹配符号串的后面增添1个符号并加入到字典中时，此时字典中可能会存在相同的短语。所以对于NGP，字典里可能会有多个相同的短语与之匹配，选择其中任何一个作为编码结果都具有相同的码字长度，并不会影响到CLZW算法的压缩性能和解压结果。匹配的多重性也是可以利用的冗余，可以在不降低压缩率的情况下，利用字典中匹配短语的多重性来携带消息。如果字典中与NGP匹配的短语数量为h，则该NGP就具有匹配的多重性h。利用NGP匹配的多重性，CLZW算法能够再携带$\lfloor\log_2 h\rfloor$ bit，而且压缩性能不会变化。

综上所述，在CLZW算法的一个编码过程中，可以携带的消息数量为$(K+L+\lfloor\log_2 h\rfloor)$ bit。CLZW压缩数据中消息比特的嵌入流程如图1所示。

3 LZW压缩数据的误码纠正

Reed-Solomon(RS)码[24]是一种基于分组的错误校正码，通常表示为RS(a, b)，其中a表示包含数据和校验码的分组长度，b表示数据的长度，RS码能在每个分组中纠正e个错误$(e=(a-b)/2)$。按照第2节的方法，设计利用CLZW算法的编码冗余携带RS码实现误码的检测和纠正。

为了能够检测和纠正LZW压缩数据中的误码，首先需要计算校验位的长度，然后通过参数K和L调整可以嵌入到压缩数据中的比特数量，以创建足够多的空间用来存放校验码。

LZW压缩数据的误码纠正流程如图2所示。压缩并嵌入校验码时按照从后到前的分组顺序，先计算数据分组$C_w(1<w\le W)$的纠错码，然后按照第2节的方法在数据分组C_{w-1}中嵌入C_w的纠错码，嵌入过程需要编码器再次压缩C_{w-1}数据分组。解压并纠正误码时按照从前往后的顺序，首先解压数据分组$C_w(1\le w<W)$，并按照第2节的方法恢复出数据分组C_{w+1}的纠错码，利用该纠错码检测和纠正C_{w+1}的错误并解压。

3.1 编码参数的设置

传递消息M时，原始数据T中用LZW和CLZW算法编码的长度分别是n_1和n_2，则有$n=n_1+n_2$。设$\bar{k}=(2^K+1)/2$, $\bar{l}=(2^L+1)/2$。根据Louchard等人[25]的研究，短语数量可表示为$s_1=n_1h_1/\log_2 n_1$ (h_1是LZW编码数据的熵)，则对应符号串的均长为$l_1=\log_2(n_1/h_1)$。能携带消息的分组数为$B=s_1/\bar{k}=n_1h_1/(\bar{k}\log_2 n_1)$，所以$|M|=(K+L)B=(K+L)n_1h_1/(\bar{k}\log_2 n_1)$。如果再利用NGP匹配的多重性，则能够携带的消息数量是$|M|=(K+L)B+\sum_{b\in B}\lfloor\log_2 h_b\rfloor=(K+L)n_1h_1/(\bar{k}\log_2 n_1)+\sum_{b\in B}\lfloor\log_2 h_b\rfloor$。NGP对应符号串的平均长度是$l_2=l_1-\bar{l}$，因此$n_2=Bl_2=s_1/\bar{k}\left(\frac{n_1}{s_1}-\bar{l}\right)=n_1/\bar{k}-\left(n_1h_1\bar{l}\right)/\left(\bar{k}\log_2 n_1\right)$。

3.2 冗余度分析

信源编码中的残留冗余，主要来自编码过程引入的冗余以及编码器忽略信源的分布特性引入的冗余[26]。由于信源编码不能完全消除信源序列中的冗余，所以可以利用冗余实现误码纠正，因此需要度量信源编码数据的冗余度。

设二进制信源符号集中，符号“0”和符号

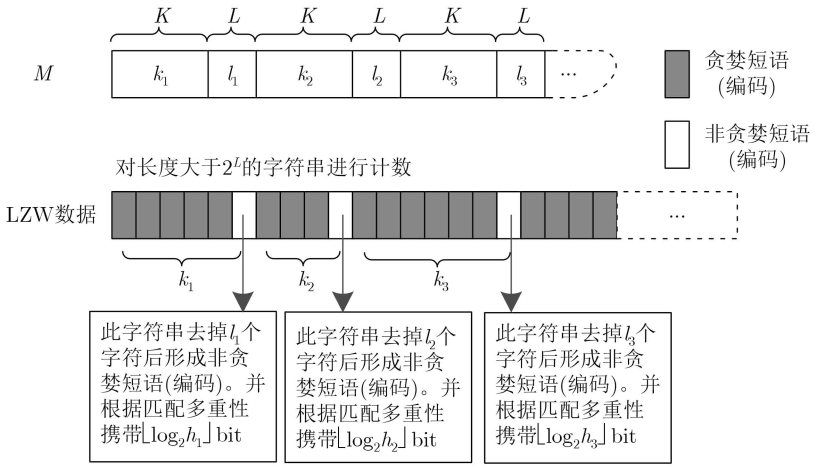

图 1 CLZW压缩数据中消息比特的嵌入

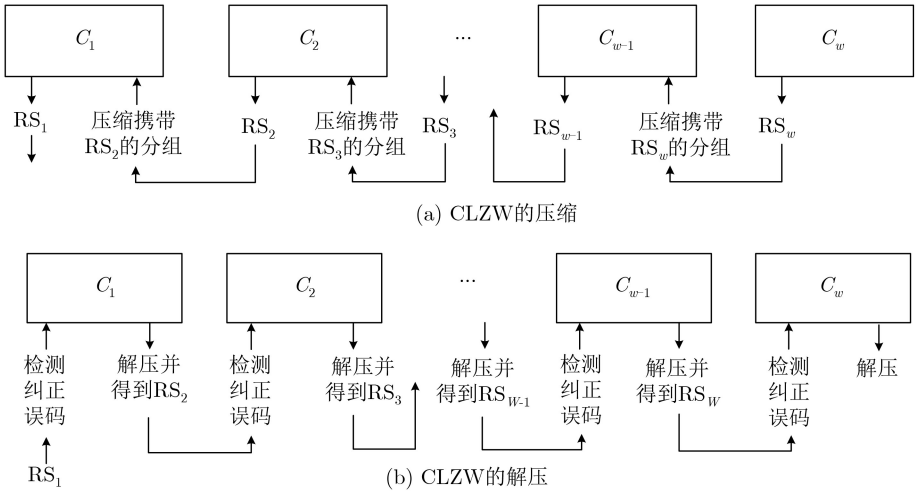

图 2 LZW压缩数据的误码纠正

“1”出现的概率分别为p和q=1–p。压缩数据时生成的码字数量为x，则数据序列长度函数$\mu(x)$可表示为

$$\mu(x)=\frac{x}{h}\lg x-\frac{Ax}{h}-\frac{x\bar{l}}{\bar{k}}+o\left(\frac{\lg x}{h}\right) \tag{1}$$

其中$A=1-\gamma-(h_2/2h)+\alpha-\delta_0(n)$，$h=-p\lg p-q\lg q>0$表示信息熵，$\gamma=0.577$, $h_2=p\lg^2 p+q\lg^2 q$, $\alpha=-\sum_{k=1}^{\infty}\frac{p^{k+1}\lg p+q^{k+1}\lg q}{1-p^{k+1}-q^{k+1}}$, $\lim_{x\to\infty}\delta_0(x)=0$。

设压缩长度为n的数据序列时，生成的码字数量为x_n，且有

$$\mu(x_n)=n \tag{2}$$

上述方程对于$n\to\infty$有渐近解[25]

$$x'_n=\frac{nh}{\lg n}\left(1+\frac{\lg\lg n}{\lg n}+\frac{A-\lg h}{\lg n}+O\left(\frac{(\lg\lg n)^2}{\lg^2 n}\right)\right) \tag{3}$$

设M_n是根据Lempel-Ziv算法解析长度为n的序列后获得的短语数量，并满足$E(M_n)\sim x_n$。

LZW算法的码字长度l_n为

$$l_n=M_n(\lg M_n+1) \tag{4}$$

LZW编码的冗余度r_s及其期望值$\overline{r_s}$分别是

$$r_s=\frac{M_n(\lg M_n+1)-nh}{n} \tag{5}$$

$$\overline{r}_s=\mathrm{E}r_n=\frac{\mathrm{E}\{M_n(\lg M_n+1)\}-nh}{n} \tag{6}$$

对于任意$k\geq1$

$$\mathrm{E}M_n^k=x_n^k\left(1+O\left(\sqrt{\frac{\lg n}{n}}\right)\right)+O\left(\frac{n^{k-1}}{\lg^{k-1}n}\right) \tag{7}$$

当$n\to\infty$时得到

$$\mathrm{E}\{M_n(\lg M_n+1)\}=\mathrm{E}M_n(\lg \mathrm{E}M_n+1)+O(1/\lg n) \tag{8}$$

令k=1，根据式(7)有$\mathrm{E}M_n=x_n\left(1+O\left(\sqrt{\lg n/n}\right)\right)$，可从式(8)得到

$$\begin{aligned}\overline{r}_s&=\frac{\mathrm{E}M_n(\lg \mathrm{E}M_n+1)-nh}{n}+O\left(\frac{1}{n\lg n}\right)\\&=\frac{x_n(\lg x_n+1)-nh}{n}+O\left(\sqrt{\frac{\lg n}{n}}\right)\\&=h\frac{2-\gamma-(h_2/2h)+\alpha-\delta_0(n)}{\lg n}+O\left(\frac{\lg\lg n}{\lg^2 n}\right)\\&=h\frac{1+A}{\lg n}+O\left(\frac{\lg\lg n}{\lg^2 n}\right)\end{aligned} \tag{9}$$

其中A是式(1)中的常数。

CLZW得到的码长为

$$L_n=x_n(1-\bar{l}/\bar{k})[\lg(x_n(1-\bar{l}/\bar{k}))+1] \tag{10}$$

因此，CLZW算法的平均冗余$\overline{r_c}$为

$$\begin{aligned}\overline{r}_c&=\frac{x_n(1-\bar{l}/\bar{k})[\lg(x_n(1-\bar{l}/\bar{k}))+1]-nh}{n}\\&=h\frac{1+A+\bar{l}/\bar{k}}{\lg n}\end{aligned} \tag{11}$$

将式(11)与式(9)进行比较，得到CLZW的平均冗余比LZW的平均冗余的增加量为

$$\Delta r=\bar{l}h/(\bar{k}\lg n) \tag{12}$$

所以，利用CLZW压缩时产生的冗余进行误码纠正是合理可行的。

4　实验及结果分析

坎特伯雷语料库(Canterbury Corpus)[27]是一组用于无损数据压缩算法基准测试的文件集合，本节使用这个语料库进行实验。实验分为2部分，第1部分测试CLZW方法的兼容能力和嵌入能力，第2部分测试CLZW方法的错误纠正能力。

4.1　兼容能力实验

实验包括CLZW压缩测试和CLZW解压测试。

压缩测试时，使用CLZW对坎特伯雷语料库中的数据进行压缩，并在压缩数据中嵌入纠错码。解压测试时，CLZW解码器接收携带消息的压缩数据，提取出携带的消息并利用纠错码对分组数据进行误码检测与纠正，再把分组数据解压生成重构文件。经过对比确认，这些重构数据和原始数据完全相同，并且从压缩数据中提取的消息与嵌入的消息一致。使用标准LZW解码器也可以正确解压出和原始数据完全相同的重构数据，证明了CLZW能够与标准LZW完全兼容，但是标准LZW解码器无法提取出压缩数据中携带的消息。

4.2 携带能力实验

分别使用LZW和CLZW压缩坎特伯雷语料库中的数据，当K=3且L=1时的部分实验结果如表1所示。实验中，使用CLZW压缩时携带的消息序列是由0和1独立同分布的模型随机生成的。$|T|$是原始数据的长度(单位：Byte); $|T'|$是LZW压缩数据的长度(单位：Byte); l是LZW压缩数据中，码字对应的符号串的平均长度；$|T'_M|$是CLZW压缩数据的长度(单位：Byte)；l_M是CLZW压缩数据中，码字对应的符号串的平均长度；$|M|$表示CLZW压缩数据能携带的消息的长度；$R=(|T'_M|-|T'|)/|T'|$表示达到和CLZW压缩数据相同的数据量时，LZW压缩数据中能够增加的消息量，与LZW压缩数据大小的比率；$R_M=|M|/|T'|$表示CLZW压缩数据中能够携带的消息量，与LZW压缩数据大小的比率。

分析实验数据可知，码字对应的符号串的平均长度越长，则压缩数据的消息携带率越高、消息携带能力越强。虽然这些原始数据的类型、内容、大小各不相同，但各个压缩数据的消息携带率相近，这表明利用LZW压缩数据携带消息的方法具有普遍通用性。

根据CLZW算法的编码规则，由于已经缩短了NGP对应符号串的长度，字典里可能会有多个相同长度的最长前缀与之匹配，选择其中任何一个都不会影响到解压结果。因此，可以进一步利用NGP最长前缀在字典中匹配的多重性来携带消息。表2给出了K=3, L=1并利用匹配的多重性时的部分实验结果。

对比表1和表2发现，利用NGP在字典中匹配的多重性来携带消息时，在压缩数据中可以携带的消息的长度$|M|$增加了。同时，由于不会改变码字的长度和其对应的符号串，所以嵌入消息后压缩数据的大小$|T'_M|$以及压缩数据中码字对应符号串的平均长度l_M不会发生变化，因此压缩性能完全不受影响。

根据表1和表2可知，每个实验数据的$|M|$都大于$|T'_M|-|T'|$，这表明在压缩后并达到相同的数据量时，CLZW压缩数据中能够携带的消息量，要多于LZW压缩数据中能够增加的消息量，即有R_M大于R。因此，当采用同种类型的纠错码且总数据量相同时，CLZW算法的纠错能力总体上优于先用LZW算法压缩再使用纠错码的性能；当采用同种类型的纠错码且纠错能力相同时，CLZW算法压缩得到的数据量总体上要少于先用LZW算法压缩再使用纠错码的数据量。

表3列出了K和L取不同值时的部分实验结果。分析实验结果发现，能够嵌入的消息长度$|M|$随着参数K的增加而逐渐减小，将L从1变为2能增加可嵌入的比特数。

表 1 分别用LZW与CLZW压缩坎特伯雷语料库的对比(K=3, L=1)

文件名	$\|T\|$	$\|T'\|$	$\|T'_M\|$	l	l_M	$\|T'_M\|-\|T'\|$	$\|M\|$	R	R_M
alice29	152089	72322	76194	3.65	3.23	3872	3982	0.053538	0.055059
cp	24603	12228	12856	3.92	3.49	628	716	0.051358	0.058554
fields	11150	5316	5580	4.11	3.66	264	322	0.049661	0.060572
ptt5	513216	70228	73961	5.78	5.30	3733	4295	0.053155	0.061158
sum	38240	31940	32605	2.49	2.17	665	1356	0.020820	0.043827

表 2 分别用LZW与CLZW压缩坎特伯雷语料库的对比

文件名	$\|T\|$	$\|T'\|$	$\|T'_M\|$	l	l_M	$\|T'_M\|-\|T'\|$	$\|M\|$	R	R_M
alice29	152089	72322	76194	3.65	3.23	3872	4113	0.053538	0.056871
cp	24603	12228	12856	3.92	3.49	628	758	0.051358	0.061989
fields	11150	5316	5580	4.11	3.66	264	331	0.049661	0.062265
ptt5	513216	70228	73961	5.78	5.30	3733	4614	0.053155	0.065700
sum	38240	31940	32605	2.49	2.17	665	1370	0.020820	0.044279

表 3　$1\leq K\leq 5$且$1\leq L\leq 2$携带消息量R_M的实验结果

文件名	L=1					L=2				
	K=1	K=2	K=3	K=4	K=5	K=1	K=2	K=3	K=4	K=5
alice29	0.081577	0.077739	0.055059	0.038675	0.024377	0.140538	0.100949	0.062275	0.037212	0.023465
cp	0.077334	0.080686	0.058554	0.040188	0.025369	0.126359	0.096884	0.058758	0.040893	0.025621
fields	0.079725	0.077587	0.060572	0.0398761	0.026660	0.116642	0.0866315	0.064385	0.040385	0.028232
ptt5	0.083042	0.080529	0.061158	0.040919	0.030843	0.130991	0.104748	0.069976	0.043431	0.030271
sum	0.073440	0.072135	0.043827	0.026355	0.018469	0.072916	0.055985	0.038270	0.029750	0.016390

4.3 错误纠正能力实验

RS码具有稳定的错误纠正能力，当采用CLZW方法传递RS码对LZW压缩数据进行误码纠正时，只要数据中的错误数不超过其纠错能力，就可以纠正压缩数据中所有的错误比特。实验中采用具有8位码元的符号表示码字，选择的校验码为RS(255, 255-2e)，数据分组的长度为255 Byte。

根据RS码的特点可知，如果错误比特的分布范围超出e符号，则无法纠正错误。根据上述分析结果可得，CLZW能够实现纠错的误比特率(Bit Error Rate: BER)满足表达式

$$\mathrm{BER}=\sum_{i=1}^{e}n_i/(a\times 8) \tag{13}$$

其中n_i表示含有误码的第$i(1\leq i\leq e)$字节中包含的错误比特数。

在实际环境中，错误比特随机分布，设数据分组中错误比特数量为k，那么含误码压缩数据的误比特率可表示为

$$\mathrm{BER}=k/(a\times 8) \tag{14}$$

当$k\leq e$时，错误比特的分布范围一定在e Byte以内，此时CLZW可以纠正数据分组中的全部错误比特。当$e<k\leq 8e$时，如果所有错误比特集中分布在e Byte内，则CLZW可以纠正全部k bit错误；如果错误比特的分布范围超出e Byte，则CLZW无法纠正错误。当$k>8e$时，错误比特的分布范围必定超出e Byte，此时CLZW无法纠正错误。

综合以上分析可知，CLZW能够纠正全部误码的BER的理论边界是

$$\mathrm{BER}_{\mathrm{AT}}=\frac{e}{a\times 8}=\frac{(a-b)/2}{8a}=\frac{a-b}{16a} \tag{15}$$

CLZW无法纠正误码的BER的理论边界是

$$\mathrm{BER}_{\mathrm{NB}}=\frac{8e}{a\times 8}=\frac{(a-b)/2}{a}=\frac{a-b}{2a} \tag{16}$$

因此，当LZW压缩数据的$\mathrm{BER}\leq\mathrm{BER}_{\mathrm{AT}}$时，采用CLZW方法可以纠正压缩数据中所有的错误比特；当$\mathrm{BER}>\mathrm{BER}_{\mathrm{NB}}$时，则CLZW无法纠正错误；当$\mathrm{BER}_{\mathrm{AT}}<\mathrm{BER}\leq\mathrm{BER}_{\mathrm{NB}}$时，采用CLZW方法能够纠正全部$k$ bit错误的概率为

$$p=\frac{C_a^e\cdot C_{8e}^k}{C_{8a}^k},\quad e<k\leq 8e \tag{17}$$

根据式(14)和式(17)，可以得到$e<k\leq 8e$时，正确纠正的概率与BER之间的关系为

$$p=\frac{C_a^e\cdot e^{8a\cdot\mathrm{BER}}}{a^{8a\cdot\mathrm{BER}}},\quad e<(8a\cdot\mathrm{BER})\leq 8e \tag{18}$$

当取e=1时，得到$\mathrm{BER}_{\mathrm{AT}}=1/(255\times 8)\approx 4.9\times 10^{-4}$，此时需要在253 Byte中传递2 Byte的校验码，即有$|M|/|T'|=2/253\approx 0.007905$。分析表3可得，在$\mathrm{BER}\leq 4.9\times 10^{-4}$的情况下，当$1\leq K\leq 5$且$1\leq L\leq 2$时，均能纠正LZW压缩数据中的所有误码。

在测试错误纠正能力时，根据参数e的不同取值，使用CLZW方法把坎特伯雷语料库中的所有文件分别压缩成LZW文件。通过引入随机分布在整个压缩数据中的不同数量的错误，来测试对误码的纠正能力。实验时，向LZW文件注入不同数量的错误进行测试，随着压缩数据中错误数量(用BER表示)的变化，对于每个文件每次确定的错误数量，进行100次具有不同随机分布错误的实验，使用CLZW方法纠正错误并解压文件，统计正确纠正并成功解压文件的概率的平均值。

图3给出了随着BER的变化，取e=1时，理论值和测量值的示意图。虚线是根据式(15)和式(18)计算得到的概率，实线是实验中测量得到的正确纠正并成功解压的平均概率。

分析图3可以发现，在BER较低时，虽然整份LZW文件中的总体错误数没有超过RS码的纠错能力，理论上可以纠正所有的错误比特，但是数据中错误比特的位置是随机的，可能会出现错误比特分布不均匀，导致个别数据分组中错误数超过了RS码的纠错能力，因此无法正确纠正并解压全部文件，且随着BER的增大，成功解压概率略有下降。一旦错误数超过了RS码的纠错能力，成功解压概率就会呈指数曲线下降。

通过实验可知，提出的无损数据压缩方案CLZW，

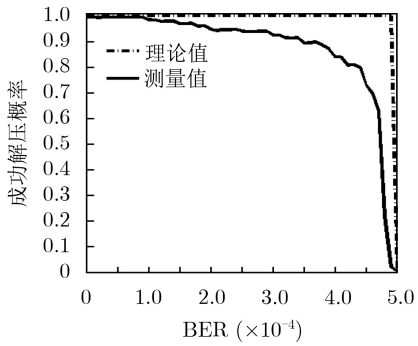

图 3 纠错率与BER的关系

通过调整压缩数据中部分符号串的长度产生的冗余来携带校验码，能够对纠错能力范围内的LZW压缩数据的误码进行有效纠正。

5 结束语

LZW字典编码是无损信源编码领域广泛应用的数据压缩方法，但LZW压缩数据中的误码极易引发码本及码字重构过程中的错误传播扩散，导致数据报废和信息丢失。为了解决LZW压缩数据的误码纠正难题，现有方案采用的是额外增加校验位保护数据的思路，这直接造成压缩效率下降，更严重的是无法兼容标准算法。为了保证实用性，本文提出的CLZW没有修改数据规格和编码方法，不需要添加校验位，因此在具有误码纠正能力的同时，与标准LZW完全兼容，而且不影响压缩性能。

参 考 文 献

[1] BERTINO E, CHOO K K R, GEORGAKOPOLOUS D, *et al*. Internet of Things (IoT): Smart and secure service delivery[J]. *ACM Transactions on Internet Technology*, 2016, 16(4): 22. doi: 10.1145/3013520.

[2] TALWANA J C and HUANG Jianhua. Smart world of Internet of Things (IoT) and its security concerns[C]. 2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Chengdu, China, 2016: 240–245. doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2016.64.

[3] WEN Lulu, ZHOU Kaile, YANG Shanlin, *et al*. Compression of smart meter big data: A survey[J]. *Renewable and Sustainable Energy Reviews*, 2018, 91: 59–69. doi: 10.1016/j.rser.2018.03.088.

[4] CHENG Ledan, GUO Songtao, WANG Ying, *et al*. Lifting wavelet compression based data aggregation in big data wireless sensor networks[C]. The 22nd IEEE International Conference on Parallel and Distributed Systems, Wuhan, China, 2016: 561–568. doi: 10.1109/ICPADS.2016.0080.

[5] 徐金甫, 刘露, 李伟, 等. 一种基于阵列配置加速比模型的无损压缩算法[J]. 电子与信息学报, 2018, 40(6): 1492–1498. doi: 10.11999/JEIT170900.
XU Jinfu, LIU Lu, LI Wei, *et al*. A new lossless compression algorithm based on array configuration speedup model[J]. *Journal of Electronics & Information Technology*, 2018, 40(6): 1492–1498. doi: 10.11999/JEIT170900.

[6] 姚军财, 刘贵忠. 一种基于人眼对比度敏感视觉特性的图像自适应量化方法[J]. 电子与信息学报, 2016, 38(5): 1202–1210. doi: 10.11999/JEIT150848.
YAO Juncai and LIU Guizhong. An adaptive quantization method of image based on the contrast sensitivity characteristics of human visual system[J]. *Journal of Electronics & Information Technology*, 2016, 38(5): 1202–1210. doi: 10.11999/JEIT150848.

[7] YANG J and BHATTACHARYA K. Combining image compression with digital image correlation[J]. *Experimental Mechanics*, 2019, 59(5): 629–642. doi: 10.1007/s11340-018-00459-y.

[8] BLASCH E, CHEN Huamei, IRVINE J M, *et al*. Prediction of compression-induced image interpretability degradation[J]. *Optical Engineering*, 2018, 57(4): 043108. doi: 10.1117/1.OE.57.4.043108.

[9] 王刚, 彭华, 唐永旺. 破损压缩文件的修复还原[J]. 电子与信息学报, 2019, 41(8): 1831–1837. doi: 10.11999/JEIT180942.
WANG Gang, PENG Hua, and TANG Yongwang. Repair and restoration of corrupted compressed files[J]. *Journal of Electronics & Information Technology*, 2019, 41(8): 1831–1837. doi: 10.11999/JEIT180942.

[10] 罗瑜, 张珍珍. 一种快速的纹理预测和混合哥伦布的无损压缩算法[J]. 电子与信息学报, 2018, 40(1): 137–142. doi: 10.11999/JEIT170305.
LUO Yu and ZHANG Zhenzhen. A fast-lossless compression using texture prediction and mixed golomb coding[J]. *Journal of Electronics & Information Technology*, 2018, 40(1): 137–142. doi: 10.11999/JEIT170305.

[11] WELCH T A. A technique for high-performance data compression[J]. *Computer*, 1984, 17(6): 8–19. doi: 10.1109/MC.1984.1659158.

[12] WANG Digang, ZHAO Xiaoqun, and SUN Qingquan. Novel fault-tolerant decompression method of corrupted huffman files[J]. *Wireless Personal Communications*, 2018, 102(4): 2555–2574. doi: 10.1007/s11277-018-5277-5.

[13] DRMOTA M and SZPANKOWSKI W. Redundancy of lossless data compression for known sources by analytic methods[J]. *Foundations and Trends® in Communications and Information Theory*, 2017, 13(4): 277–417. doi: 10.1561/0100000090.

[14] KOGA H and YAMAMOTO H. Asymptotic properties on codeword lengths of an optimal FV code for general sources[J]. *IEEE Transactions on Information Theory*, 2005, 51(4): 1546–1555. doi: 10.1109/TIT.2005.844098.

[15] FRENKEL S, KOPEETSKY M, and MOLOTKOVSKI R. Lempel-Ziv-welch compression algorithm with exponential decay[C]. The 2nd International Symposium on Stochastic Models in Reliability Engineering, Life Science and Operations Management, Beer-Sheva, Israel, 2016: 616–619. doi: 10.1109/SMRLO.2016.108.

[16] 李从鹤, 郑辉. 一种用于文本压缩的信源容错译码算法[J]. 无线电通信技术, 2006, 32(2): 36–38, 64. doi: 10.3969/j.issn.1003-3114.2006.02.013.
LI Conghe and ZHENG Hui. A fault-tolerance decoding algorithm for text compression[J]. *Radio Communications Technology*, 2006, 32(2): 36–38, 64. doi: 10.3969/j.issn.1003-3114.2006.02.013.

[17] KLEIN S T and SHAPIRA D. Practical fixed length Lempel-Ziv coding[J]. *Discrete Applied Mathematics*, 2014, 163: 326–333. doi: 10.1016/j.dam.2013.08.022.

[18] ZHANG Jie, YANG Enhui, and KIEFFER J C. A universal grammar-based code for lossless compression of binary trees[J]. *IEEE Transactions on Information Theory*, 2014, 60(3): 1373–1386. doi: 10.1109/TIT.2013.2295392.

[19] KWON B, GONG M, and LEE S. Novel error detection algorithm for LZSS compressed data[J]. *IEEE Access*, 2017, 5: 8940–8947. doi: 10.1109/ACCESS.2017.2704900.

[20] KITAKAMI M and KAWASAKI T. Burst error recovery method for LZSS coding[J]. *IEICE Transactions on Information and Systems*, 2009, 92(12): 2439–2444. doi: 10.1587/transinf.e92.d.2439.

[21] PEREIRA Z C, PELLENZ M E, SOUZA R D, *et al*. Unequal error protection for LZSS compressed data using Reed-Solomon codes[J]. *IET Communications*, 2007, 1(4): 612–617. doi: 10.1049/iet-com:20060530.

[22] KEMPA D and KOSOLOBOV D. LZ-end parsing in compressed space[C]. 2017 Data Compression Conference, Snowbird, USA, 2017: 350-359.

[23] DO H H, JANSSON J, SADAKANE K, *et al*. Fast relative Lempel-Ziv self-index for similar sequences[J]. *Theoretical Computer Science*, 2014, 532: 14–30. doi: 10.1016/j.tcs.2013.07.024.

[24] REED I S and SOLOMON G. Polynomial codes over certain finite fields[J]. *Journal of the Society for Industrial and Applied Mathematics*, 1960, 8(2): 300–304. doi: 10.1137/0108018.

[25] LOUCHARD G and SZPANKOWSKI W. On the average redundancy rate of the Lempel-Ziv code[J]. *IEEE Transactions on Information Theory*, 1997, 43(1): 2–8. doi: 10.1109/18.567640.

[26] DAS S, BULL D M, and WHATMOUGH P N. Error-resilient design techniques for reliable and dependable computing[J]. *IEEE Transactions on Device and Materials Reliability*, 2015, 15(1): 24–34. doi: 10.1109/tdmr.2015.2389038.

[27] The Canterbury corpus[EB/OL]. http://corpus.canterbury.ac.nz/descriptions/#cantrbry, 2018.

王　刚：男，1981年生，副教授，研究方向为信号分析、信息处理、模式识别.

靳彦青：女，1983年生，工程师，研究方向为移动通信.

彭　华：男，1973年生，教授，研究方向为通信信号处理、软件无线电.

张光伟：男，1984年生，讲师，研究方向为信息安全.